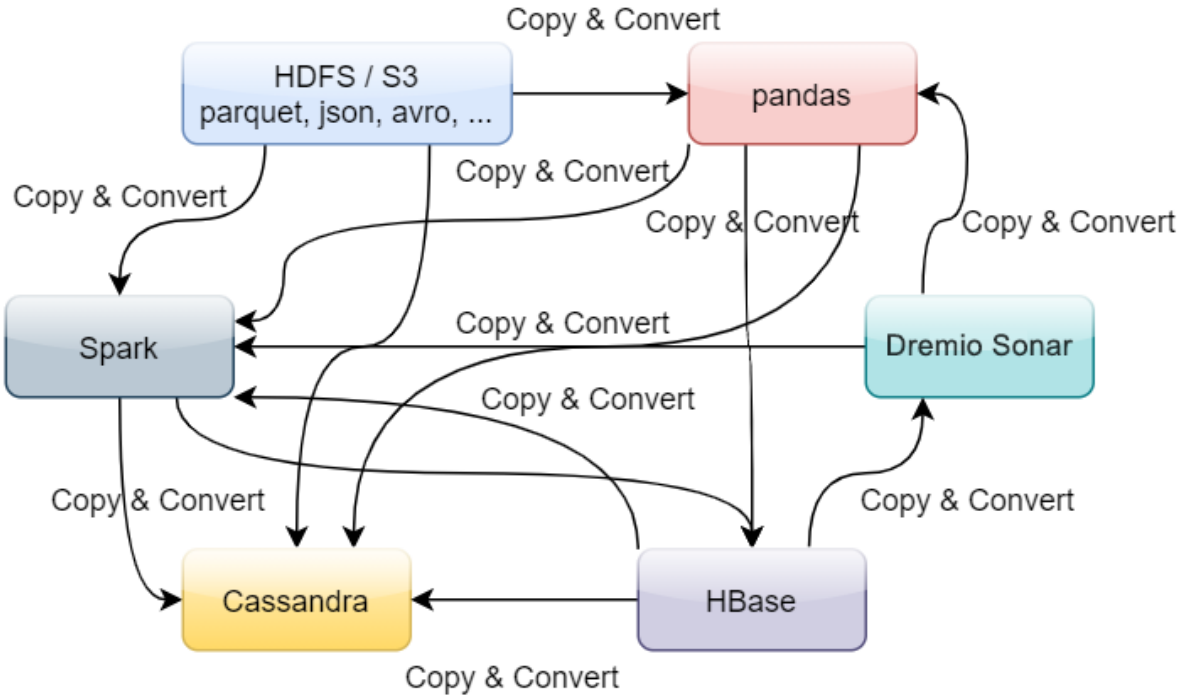
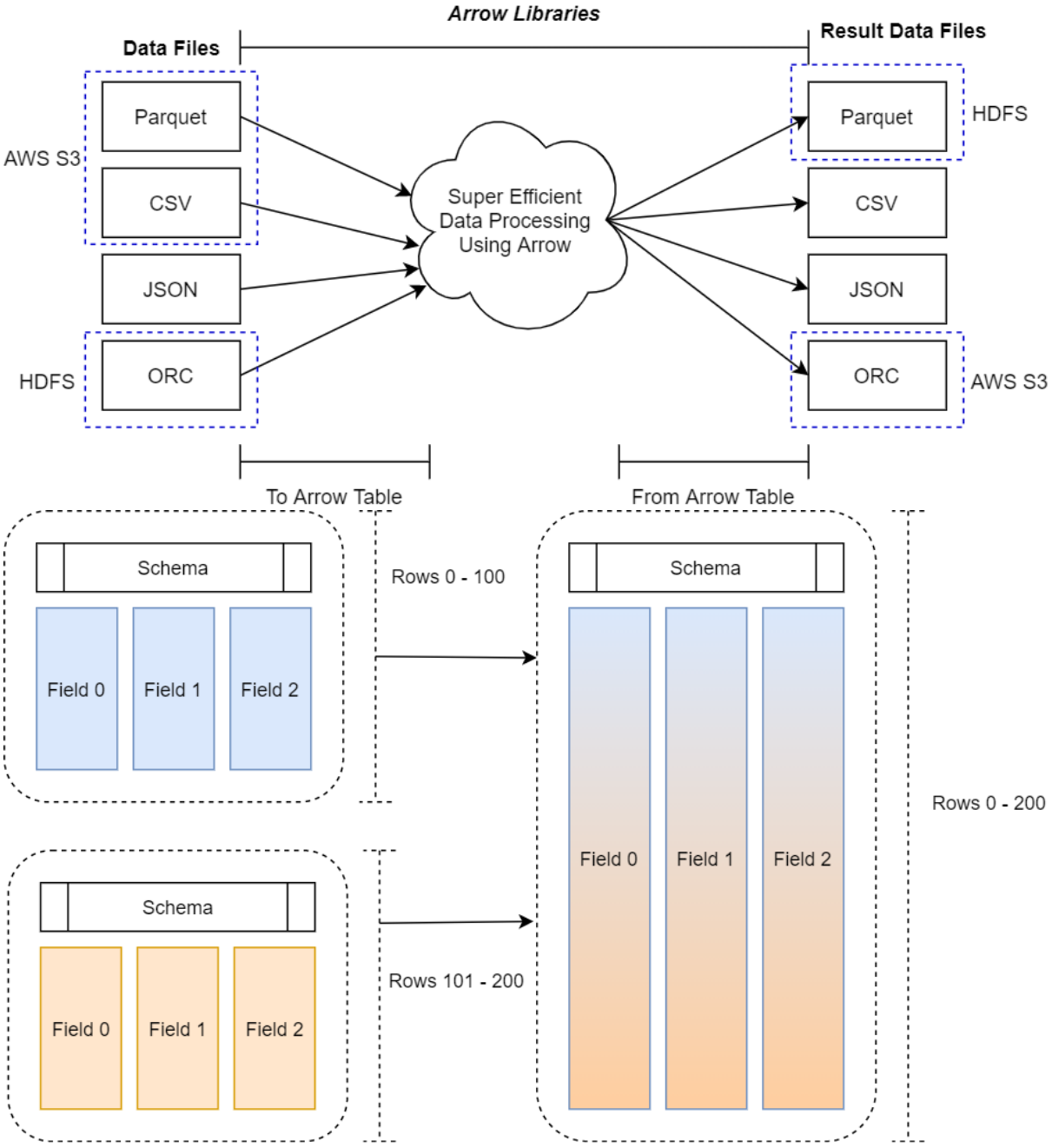
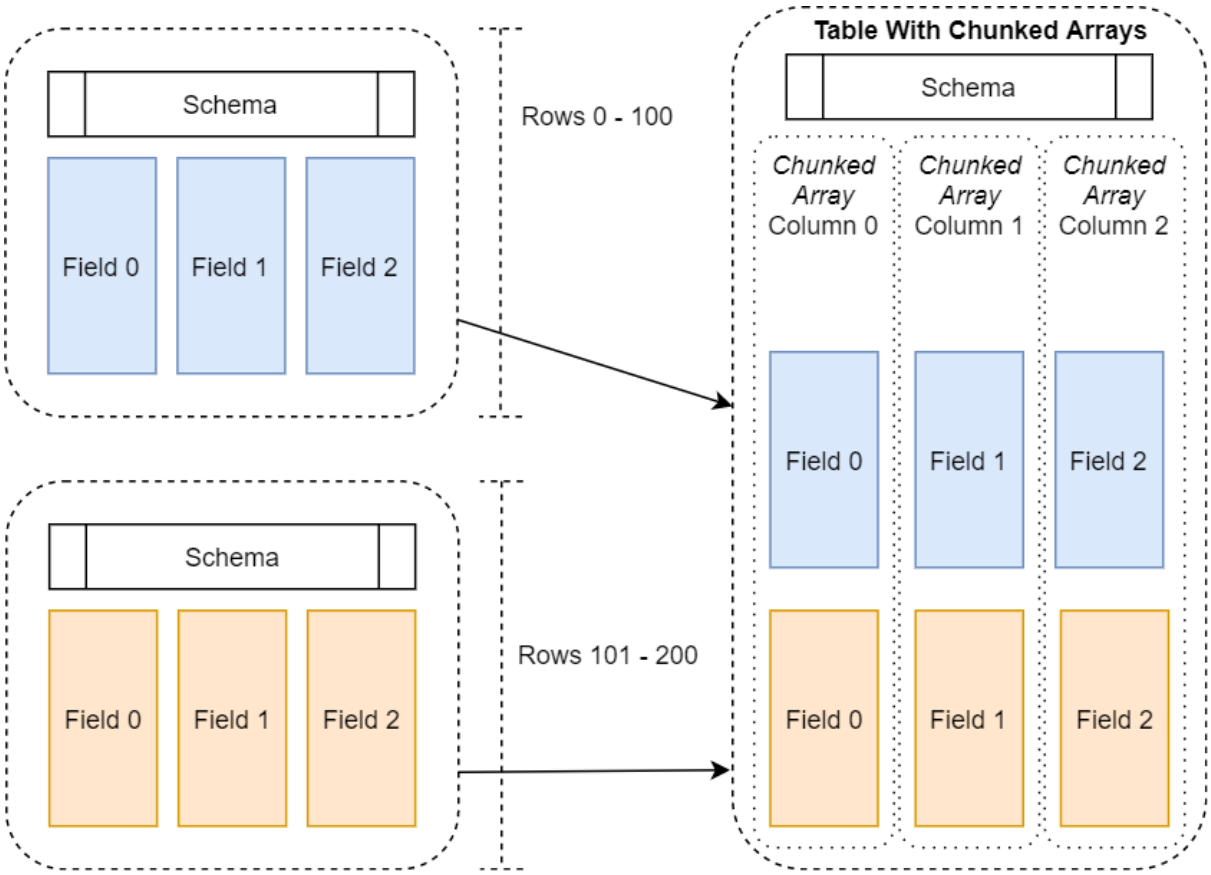
Chapter 1: Getting Started with Apache Arrow

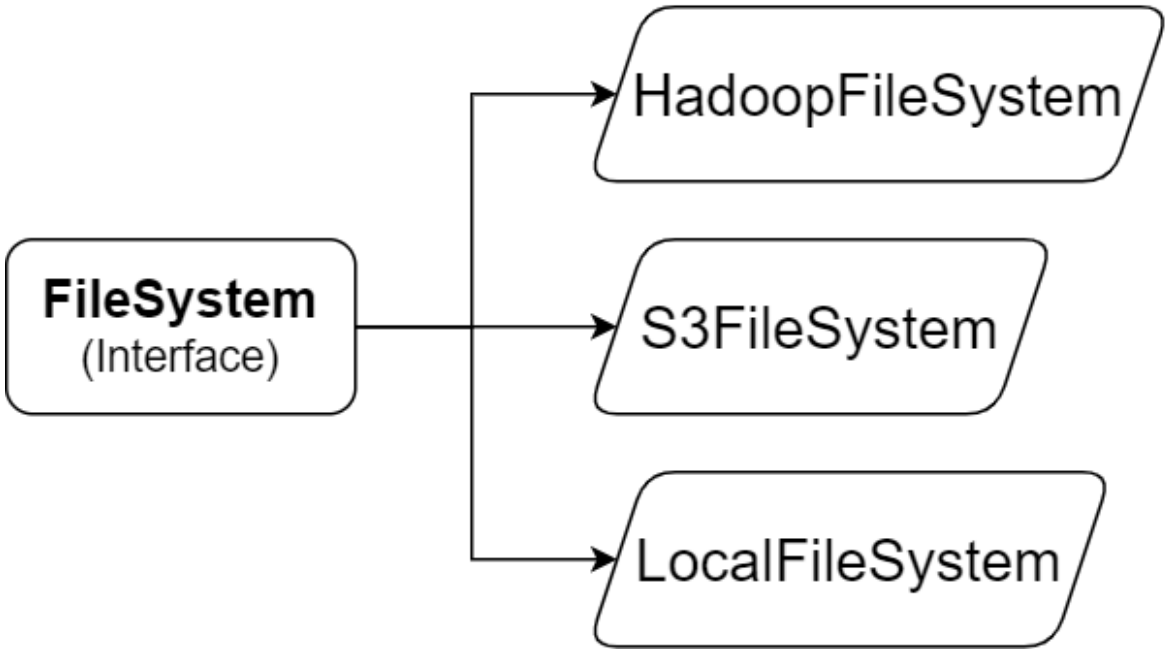


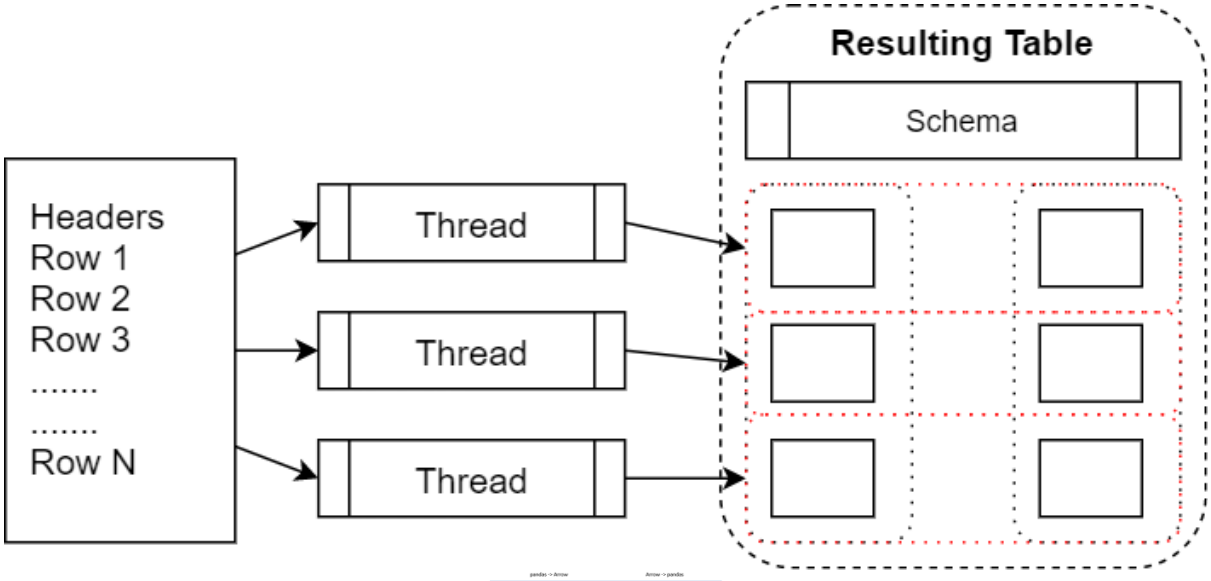
Chapter 2: Working with Key Arrow Specifications



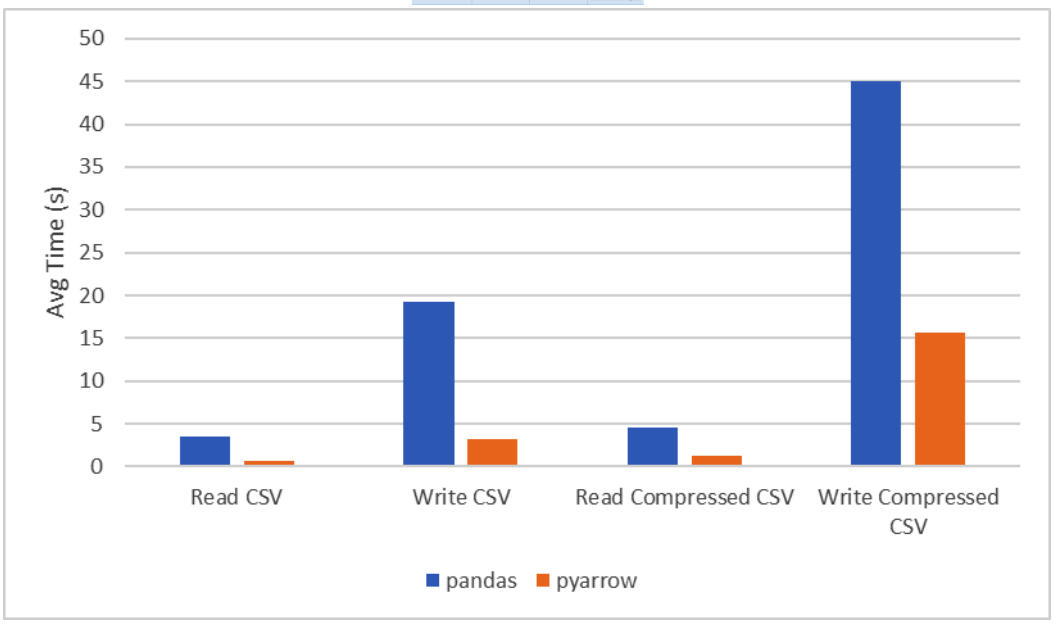


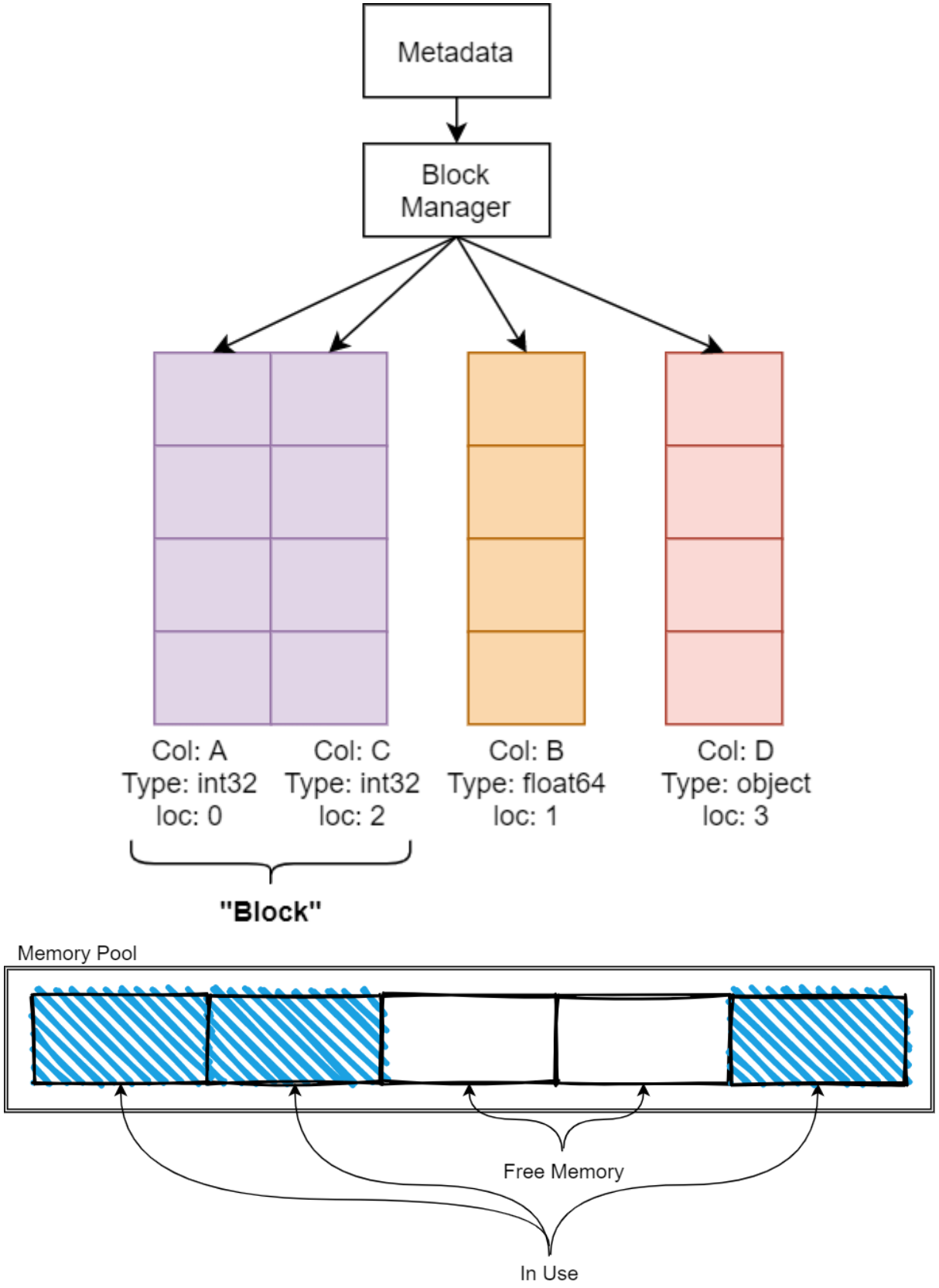
Concrete Implementations

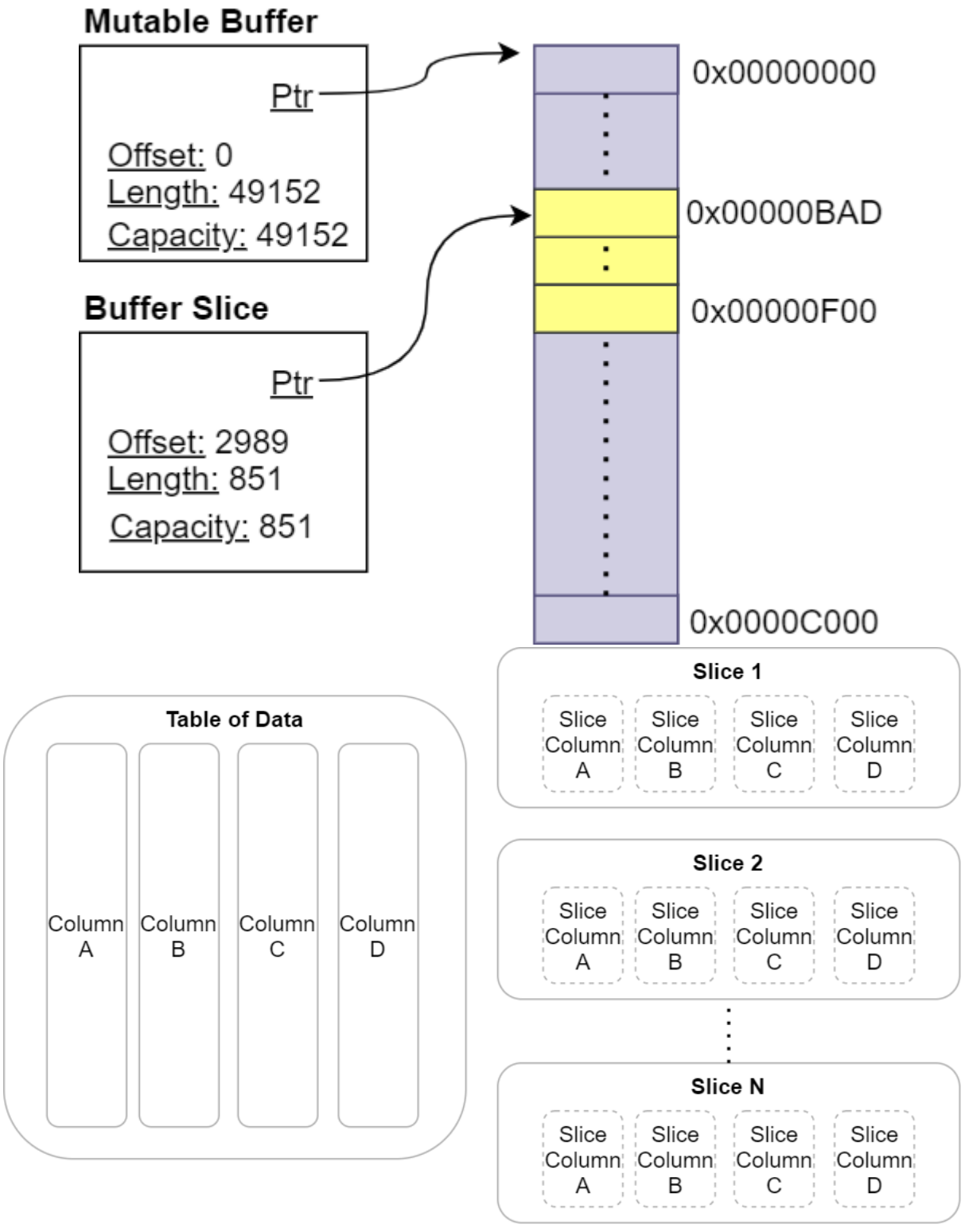




pandas -> Arrow		Arrow -> pandas	
pandas (source)	Arrow (destination)	Arrow (source)	pandas (destination)
bool	BOOL	BOOL	bool
int or uint {8,16,32,64}	int or uint {8,16,32,64}	BOOL with nulls	object with values True, False, None
float32	FLOAT	int or uint {8,16,32,64}	int or uint {8,16,32,64}
float64	DOUBLE	int or uint {8,16,32,64} with nulls	float64
str/unicode	STRING	FLOAT	float32
pandas.Categorical	DICTIONARY	DOUBLE	float64
pandas.Timestamp	TIMESTAMP(unit=ns)	STRING	str
datetime.date	DATE	DICTIONARY	pandas.Categorical
datetime.time	TIME64	TIMESTAMP(unit='*')	Pandas.Timestamp[numpy.datetime64[ns]]
		DATE	object with datetime.date objects




Metadata
Block Manager
Col: A
Type: int32
loc: 0
Col: C
Type: int32
loc: 2
Col: B
Type: float64
loc: 1
Col: D
Type: object
loc: 3
"Block"
Memory Pool
Free Memory
In Use


Mutable Buffer
Ptr
Offset: 0
Length: 49152
Capacity: 49152
Buffer Slice
Ptr
Offset: 2989
Length: 851
Capacity: 851
0x00000000
0x00000BAD
0x00000F00
0x0000C000
Table of Data
Column A
Column B
Column C
Column D
Slice 1
Slice Column A
Slice Column B
Slice Column C
Slice Column D
Slice 2
Slice Column A
Slice Column B
Slice Column C
Slice Column D
Slice N
Slice Column A
Slice Column B
Slice Column C
Slice Column D

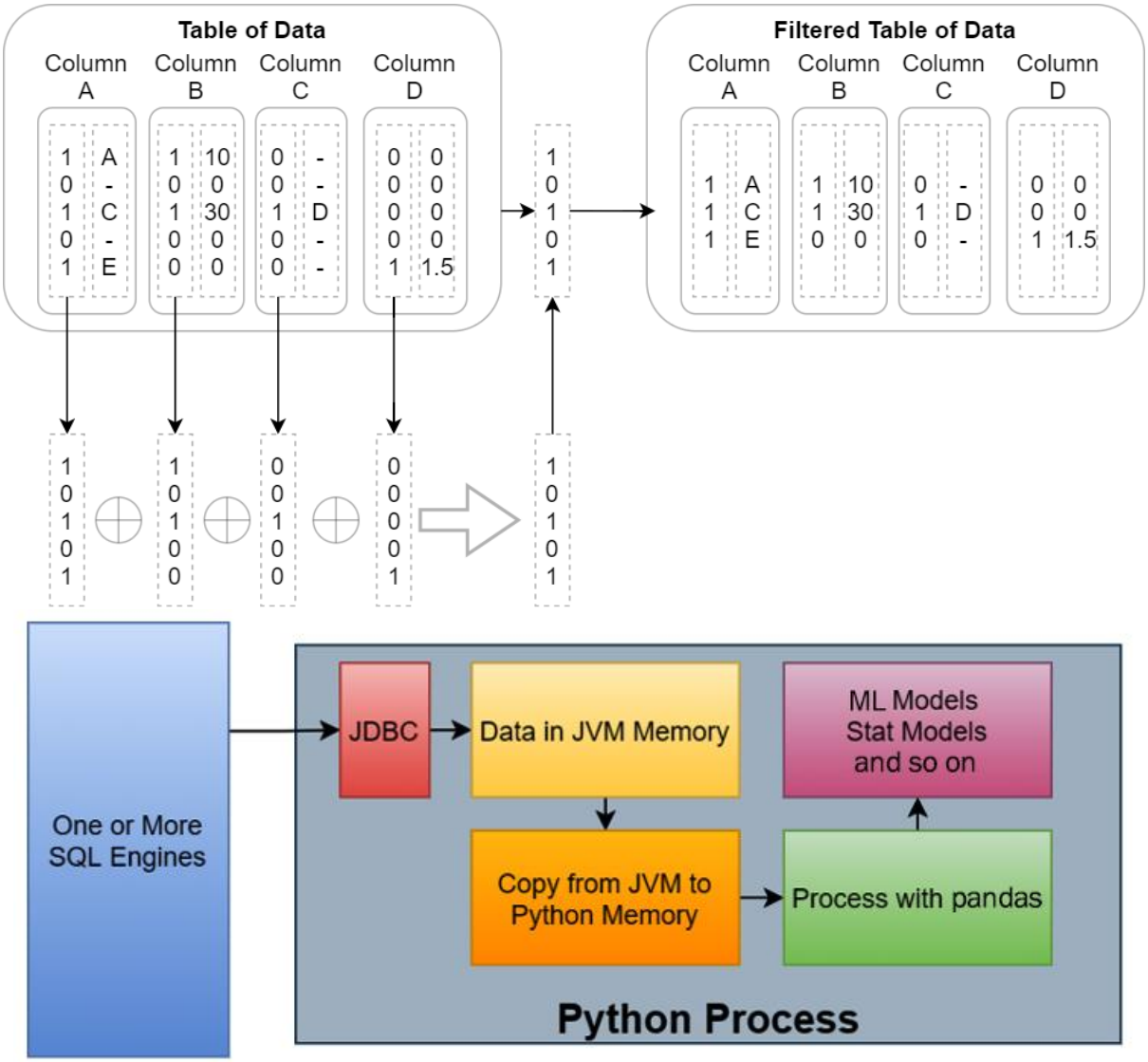
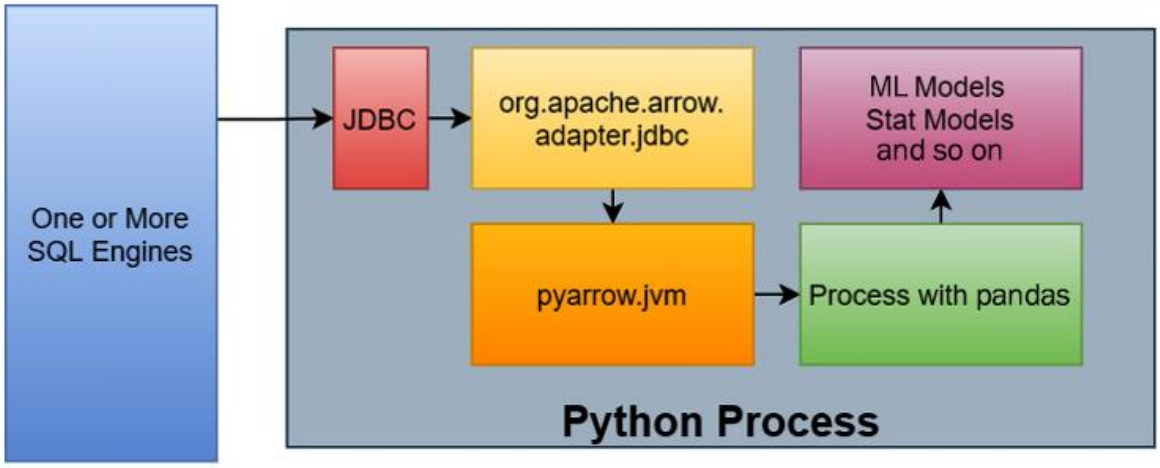
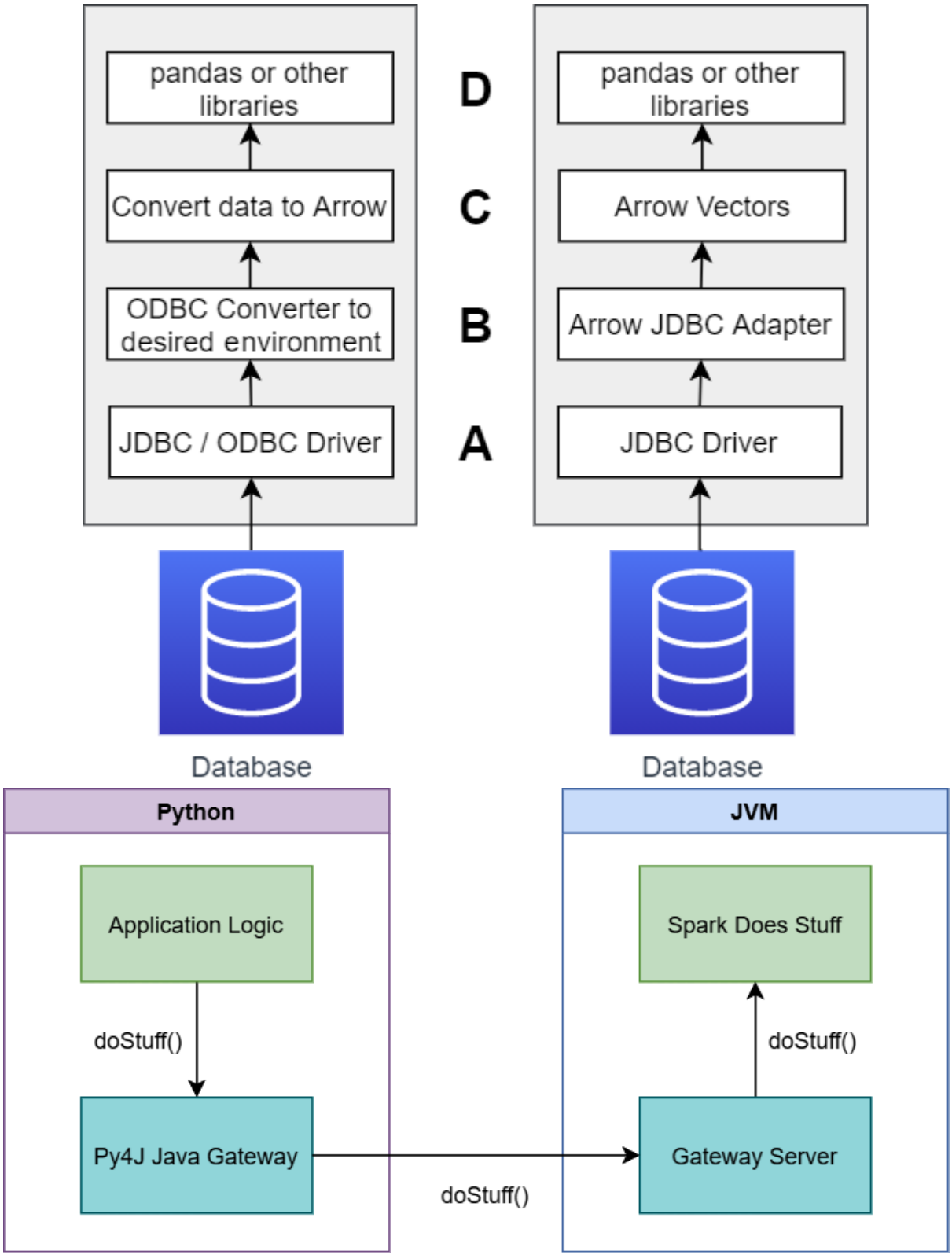


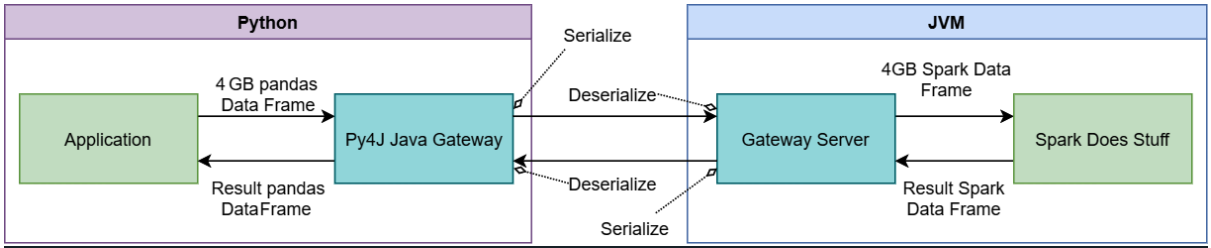
Table of Data
Column A
Column B
Column C
Column D
Filtered Table of Data
Column A
Column B
Column C
Column D
One or More SQL Engines
JDBC
Data in JVM Memory
ML Models
Stat Models
and so on
Copy from JVM to Python Memory
Process with pandas
Python Process



# Rows	Traditional (Copy)	Shared Memory (No Copy)	Speedup	% Improvement
950	274 ms	144 ms	1.90x	90.27%
10,000	1.29 s	175 ms	7.37x	637.14%
198,143	26.8 s	403 ms	66.50x	6550.12%
623,418	1 min 5 s	573 ms	113.44x	11243.80%

Chapter 3: Data Science with Apache Arrow





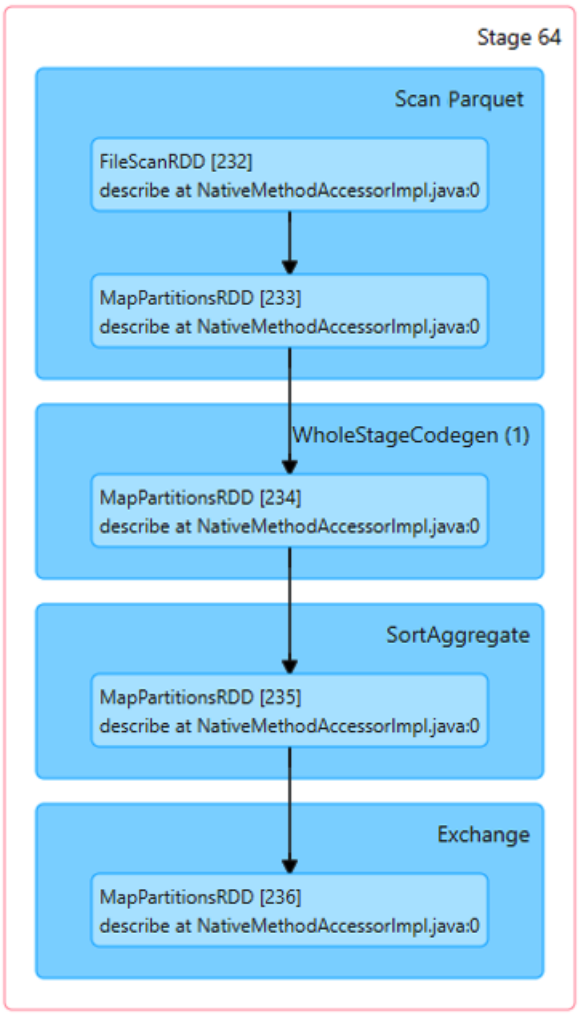

```
[I 2021-11-
15 04:24:34.386 ServerApp]□(B Writing Jupyter server cookie secret to /home/jovyan/.local/share/jupyter/runtime/jupyter_cookie_secret
[I 2021-11-15 04:24:34.605 ServerApp]□(B nbclassic | extension was successfully linked.
[I 2021-11-15 04:24:34.637 ServerApp]□(B nbclassic | extension was successfully loaded.
[I 2021-11-15 04:24:34.638 LabApp]□(B JupyterLab extension loaded from /opt/conda/lib/python3.9/site-packages/jupyterlab
[I 2021-11-15 04:24:34.638 LabApp]□(B JupyterLab application directory is /opt/conda/share/jupyter/lab
[I 2021-11-15 04:24:34.643 ServerApp]□(B jupyterlab | extension was successfully loaded.
[I 2021-11-15 04:24:34.644 ServerApp]□(B Serving notebooks from local directory: /home/jovyan
[I 2021-11-15 04:24:34.644 ServerApp]□(B Jupyter Server 1.11.2 is running at:
[I 2021-11-15 04:24:34.644 ServerApp]□(B http://2dd5d89e795a:8888/lab?token=a58056e05bf0e3b751c087689952ec5776085702347b1fe6
[I 2021-11-15 04:24:34.644 ServerApp]□(B  or http://127.0.0.1:8888/lab?token=a58056e05bf0e3b751c087689952ec5776085702347b1fe6
[I 2021-11-15 04:24:34.644 ServerApp]□(B Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 2021-11-15 04:24:34.650 ServerApp]□(B

    To access the server, open this file in a browser:
        file:///home/jovyan/.local/share/jupyter/runtime/jpserver-8-open.html
    Or copy and paste one of these URLs:
        http://2dd5d89e795a:8888/lab?token=a58056e05bf0e3b751c087689952ec5776085702347b1fe6
     or http://127.0.0.1:8888/lab?token=a58056e05bf0e3b751c087689952ec5776085702347b1fe6
```

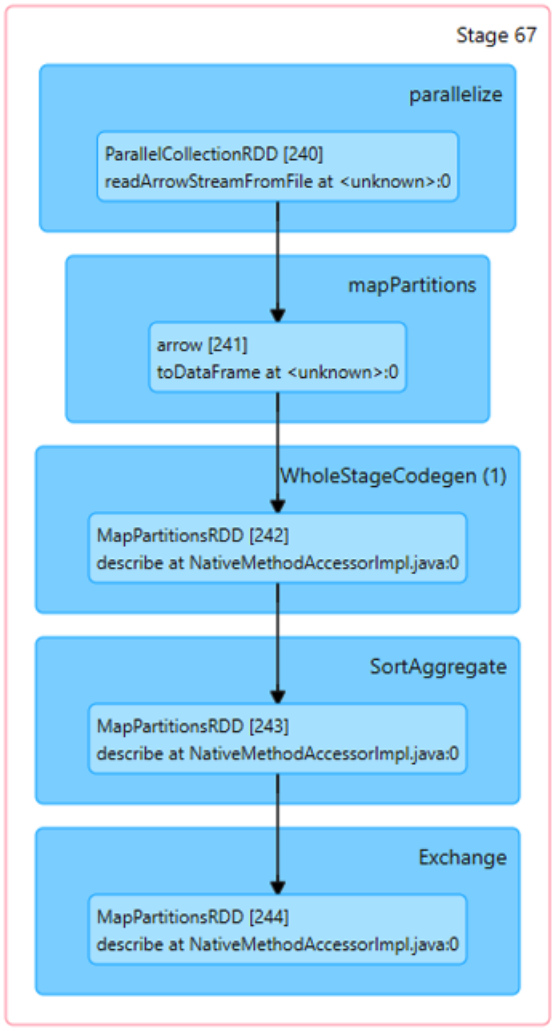
chapter3.ipynb

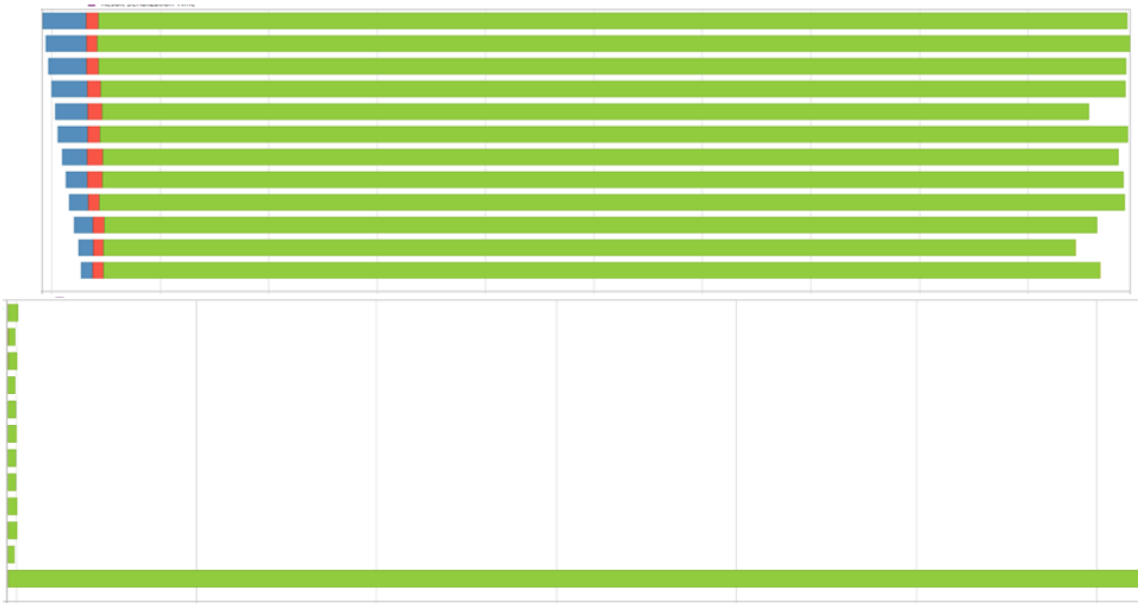
Code

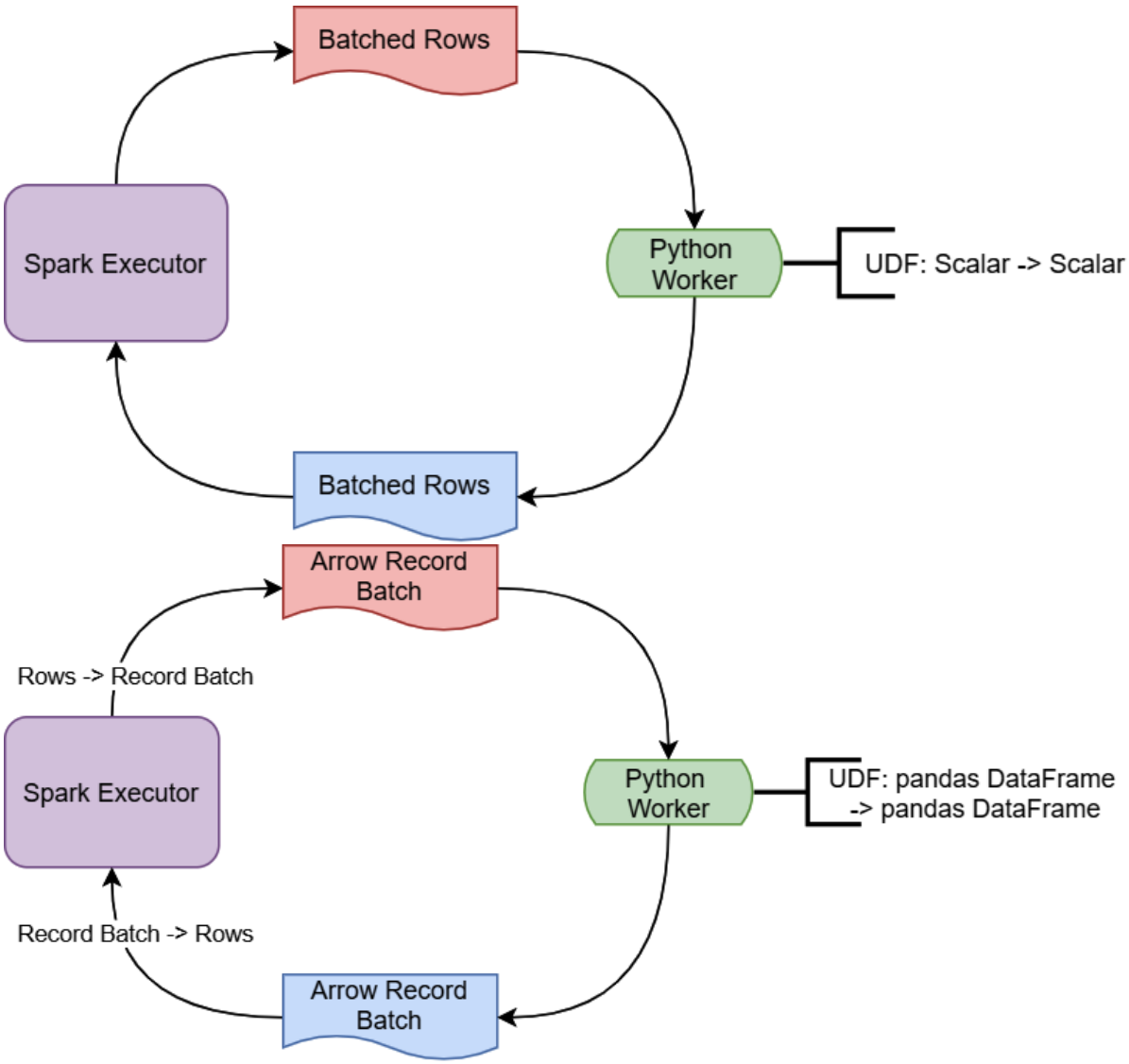
```
•[1]: import pyspark as pyspark
      from pyspark import SparkContext
      conf = pyspark.SparkConf()
      conf.set("spark.executor.memory", "8g")
      conf.set("spark.driver.memory", "8g")
      conf.set('spark.jars.packages', 'org.apache.hadoop:hadoop-aws:3.3.1')
      sc = SparkContext(conf=conf)
      sc._jsc.hadoopConfiguration().set('fs.s3a.aws.credentials.provider', 'org.apache.hadoop.fs.s3a.AnonymousAWSCredentialsProvider')
      sc._jsc.hadoopConfiguration().set('fs.s3a.impl', 'org.apache.hadoop.fs.s3a.S3AFileSystem')

      from pyspark.sql import SparkSession
      spark = SparkSession(sc)
```

```
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/usr/local/spark-3.2.0-bin-hadoop3.2/jars/spark-unsafe
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
:: loading settings :: url = jar:file:/usr/local/spark-3.2.0-bin-hadoop3.2/jars/ivy-2.5.0.jar!/org/apache/ivy/core/settings/ivysett
Ivy Default Cache set to: /home/jovyan/.ivy2/cache
```


Stage 64
Scan Parquet
FileScanRDD [232]
describe at NativeMethodAccessorImpl.java:0
MapPartitionsRDD [233]
describe at NativeMethodAccessorImpl.java:0
WholeStageCodegen (1)
MapPartitionsRDD [234]
describe at NativeMethodAccessorImpl.java:0
SortAggregate
MapPartitionsRDD [235]
describe at NativeMethodAccessorImpl.java:0
Exchange
MapPartitionsRDD [236]
describe at NativeMethodAccessorImpl.java:0


Stage 67
parallelize
ParallelCollectionRDD [240]
readArrowStreamFromFile at <unknown>:0
mapPartitions
arrow [241]
toDataFrame at <unknown>:0
WholeStageCodegen (1)
MapPartitionsRDD [242]
describe at NativeMethodAccessorImpl.java:0
SortAggregate
MapPartitionsRDD [243]
describe at NativeMethodAccessorImpl.java:0
Exchange
MapPartitionsRDD [244]
describe at NativeMethodAccessorImpl.java:0

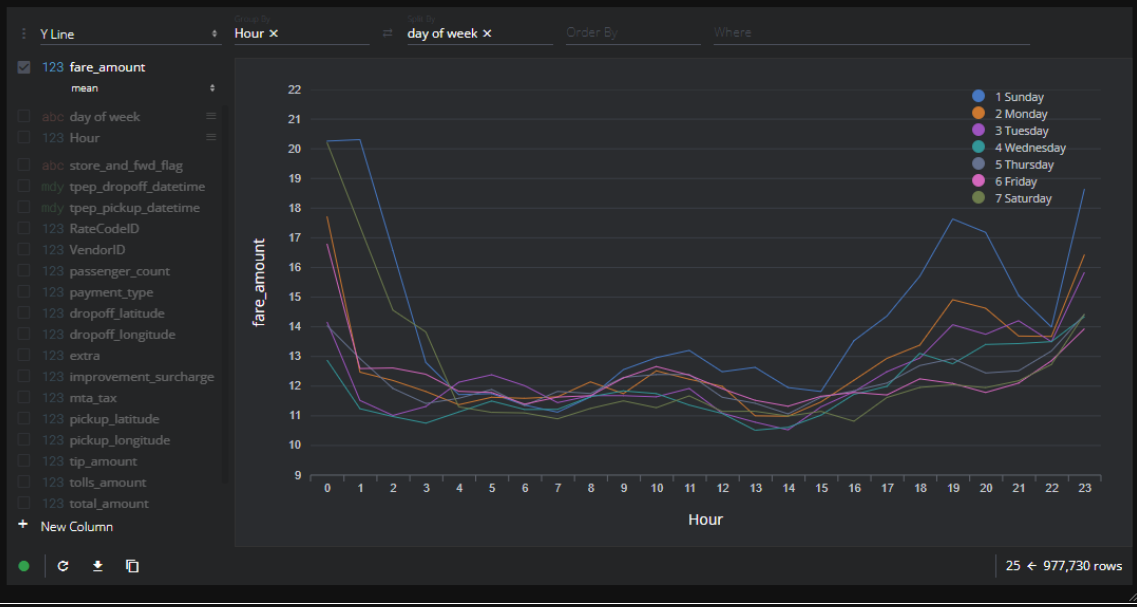



Batched Rows
Spark Executor
Python Worker
UDF: Scalar -> Scalar
Batched Rows
Arrow Record Batch
Rows -> Record Batch
Spark Executor
Python Worker
UDF: pandas DataFrame -> pandas DataFrame
Record Batch -> Rows
Arrow Record Batch

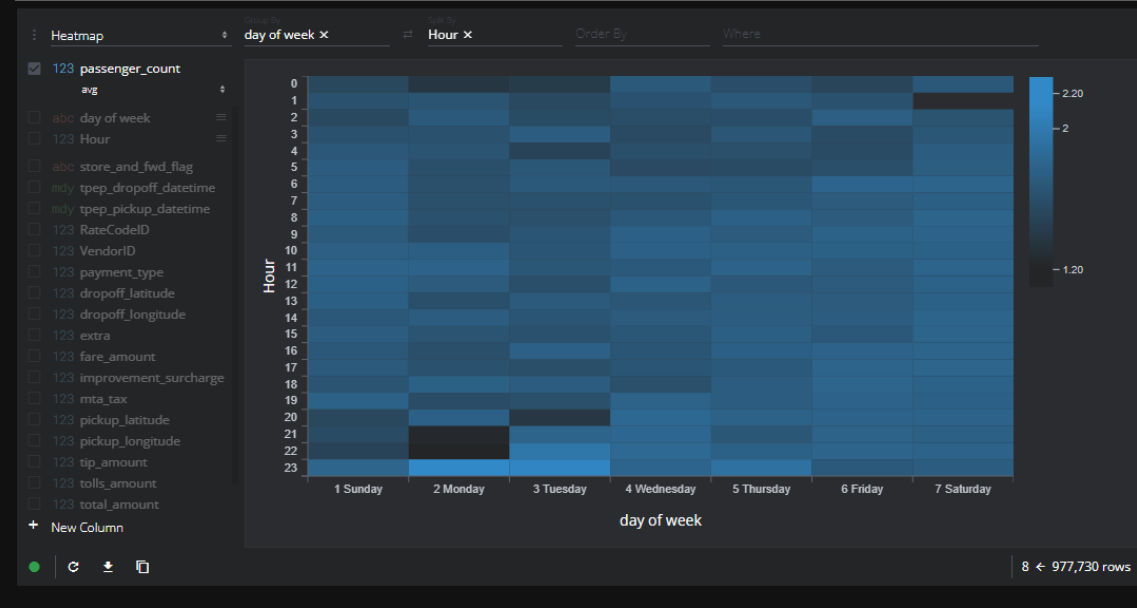
[8]:
```
view = table.view(filter=[["tpep_pickup_datetime", "<", "2015-01-10"]])
display(view.num_rows())
```

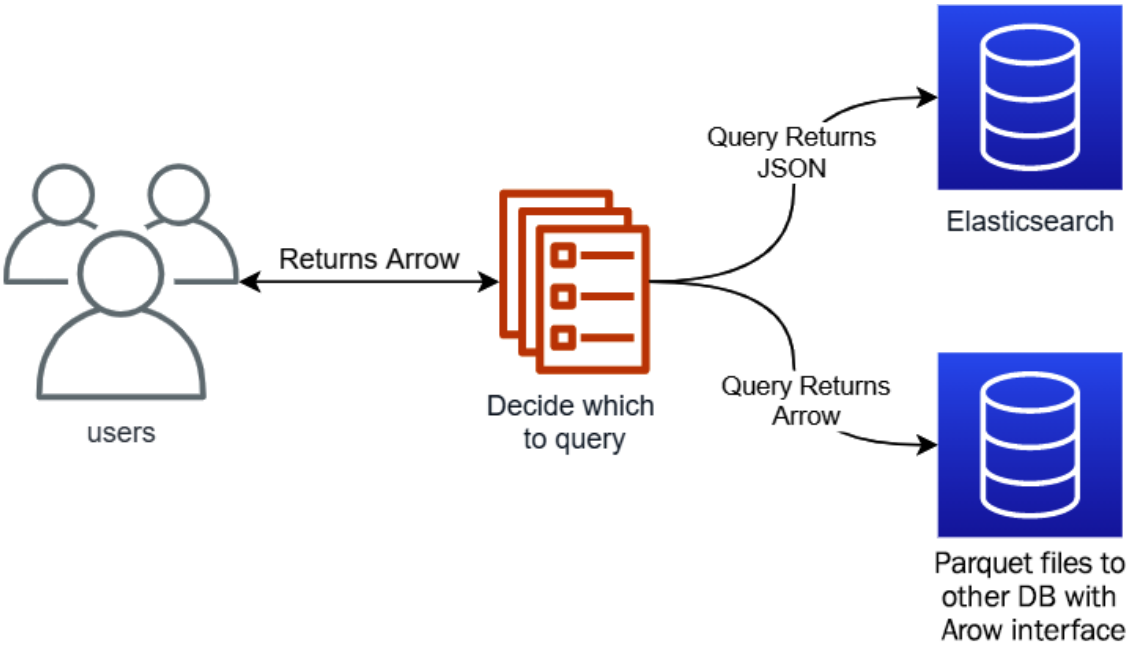
```
977730
```

[10]:
```
widget = PerspectiveWidget(view.to_arrow())
widget
```

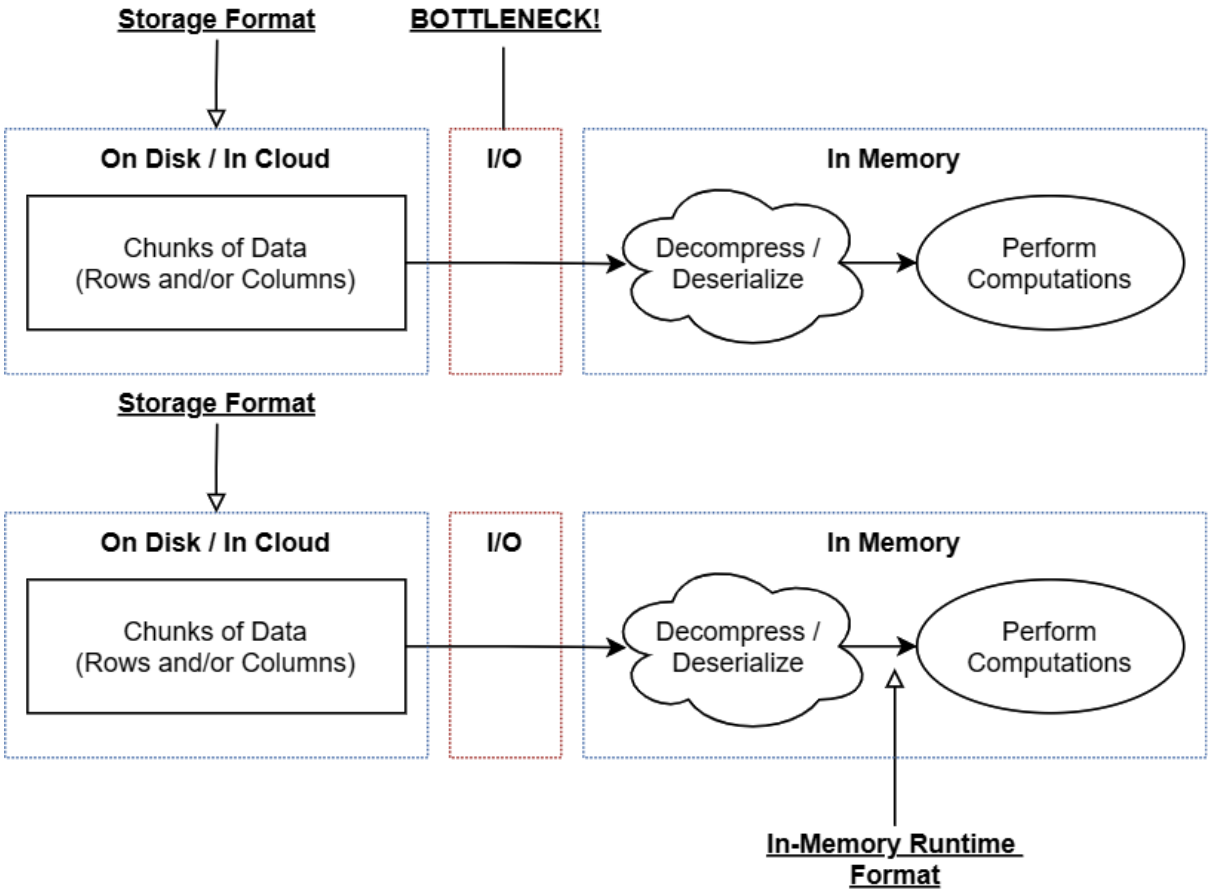


[10]:
```
widget = PerspectiveWidget(view.to_arrow())
widget
```

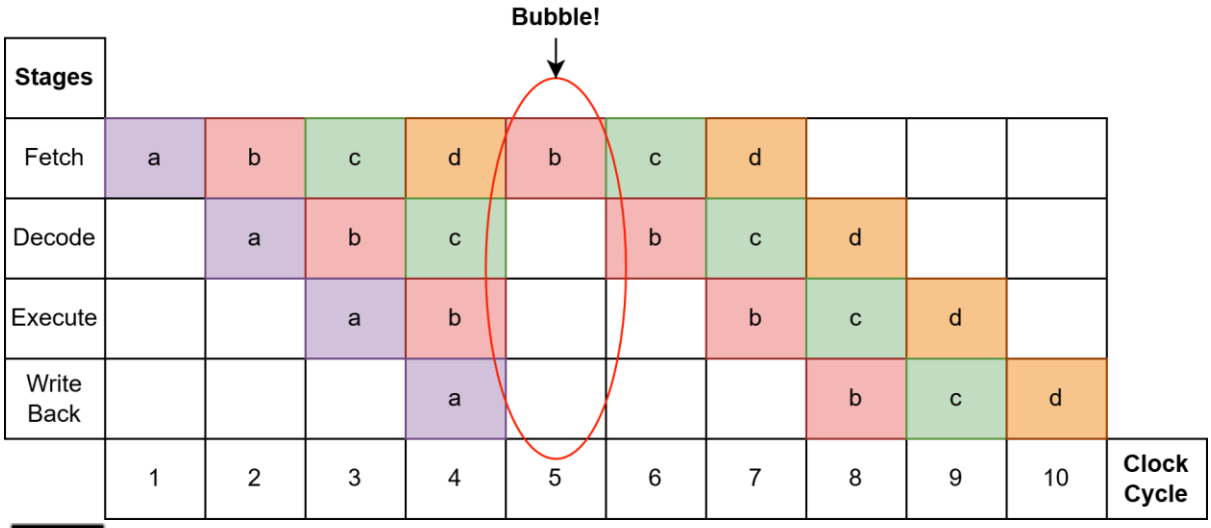


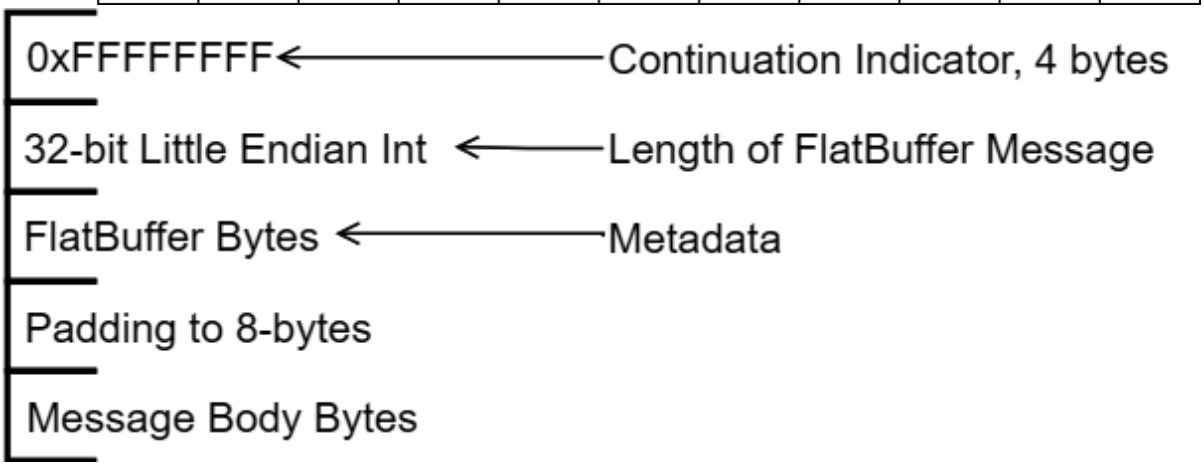

Query Returns
JSON
Elasticsearch
Returns Arrow
Decide which
to query
Query Returns
Arrow
users
Parquet files to
other DB with
Arow interface

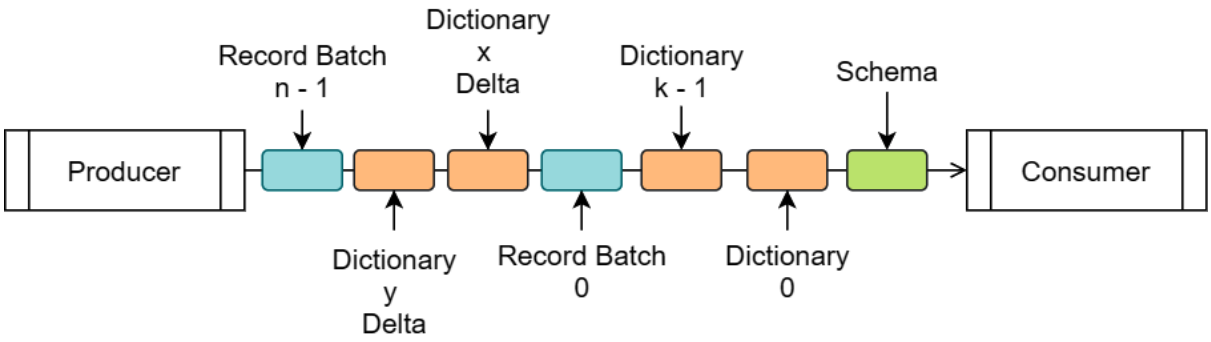
Chapter 4: Format and Memory Handling

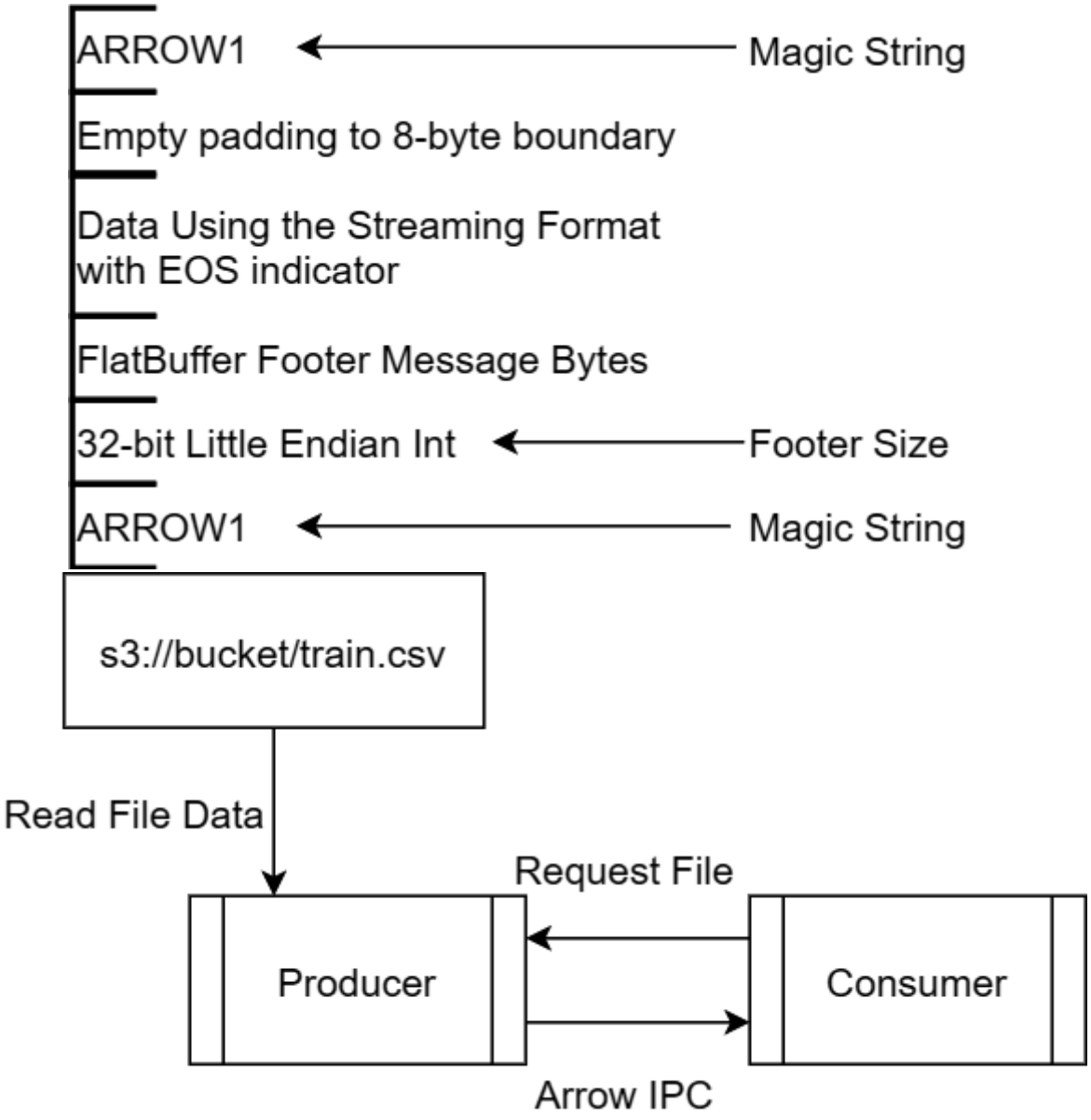


Stages											
Fetch	a	b	c	d							
Decode		a	b	c	d						
Execute			a	b	c	d					
Write Back				a	b	c	d				
	1	2	3	4	5	6	7	8	9	10	Clock Cycle


Bubble!
Stages
Fetch
Decode
Execute
Write Back
a
b
c
d
1
2
3
4
5
6
7
8
9
10
Clock Cycle


0xFFFFFFFF
Continuation Indicator, 4 bytes
32-bit Little Endian Int
Length of FlatBuffer Message
FlatBuffer Bytes
Metadata
Padding to 8-bytes
Message Body Bytes

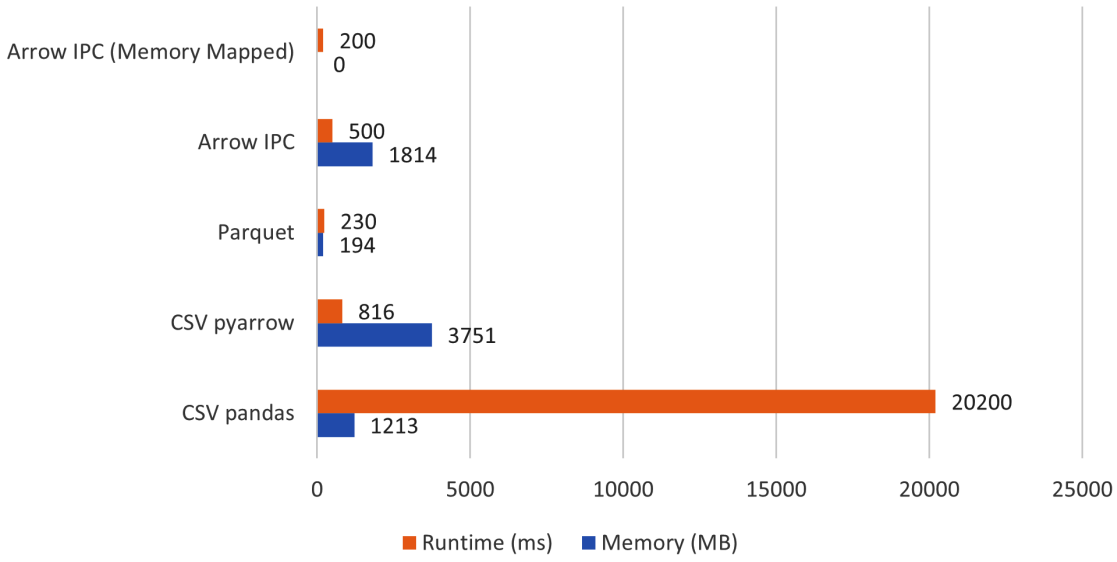

Record Batch n - 1
Dictionary x Delta
Dictionary k - 1
Schema
Producer
Consumer
Dictionary y Delta
Record Batch 0
Dictionary 0

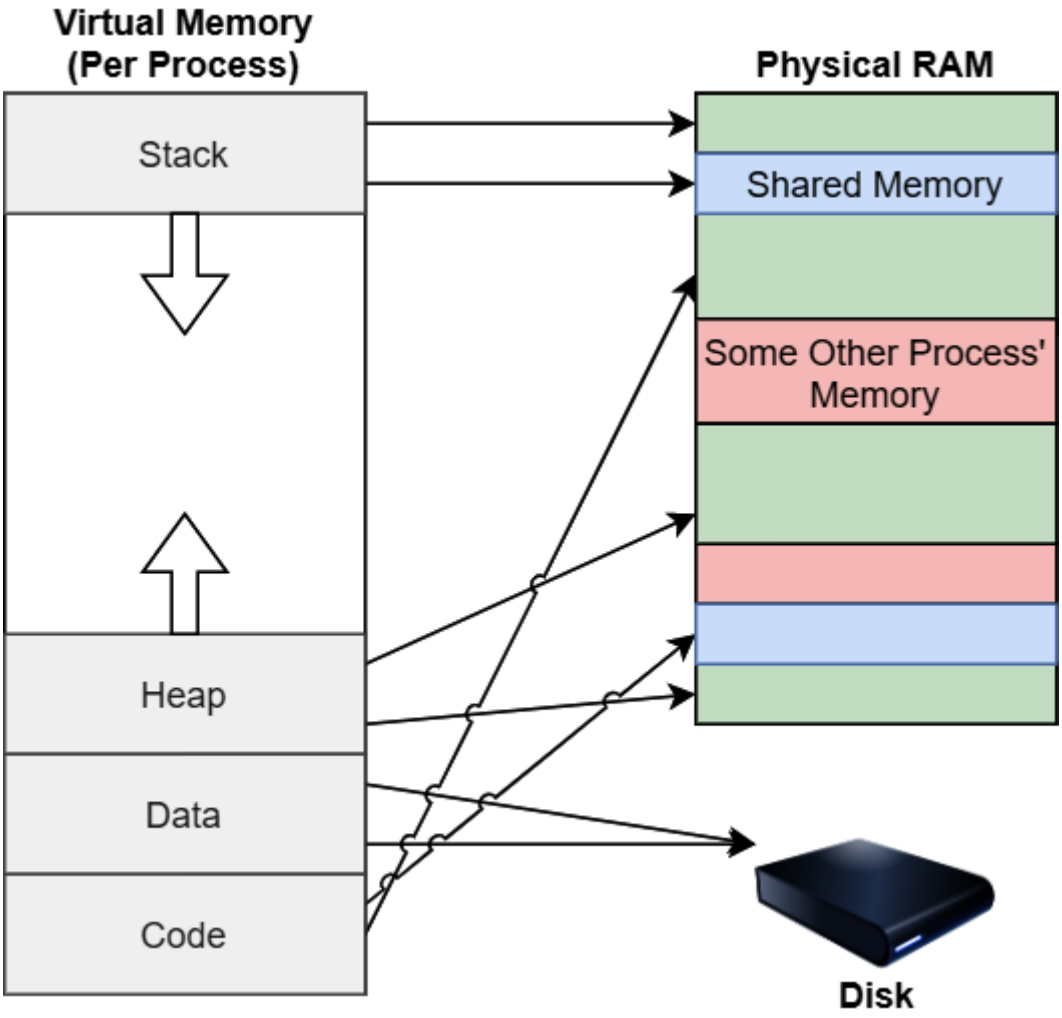


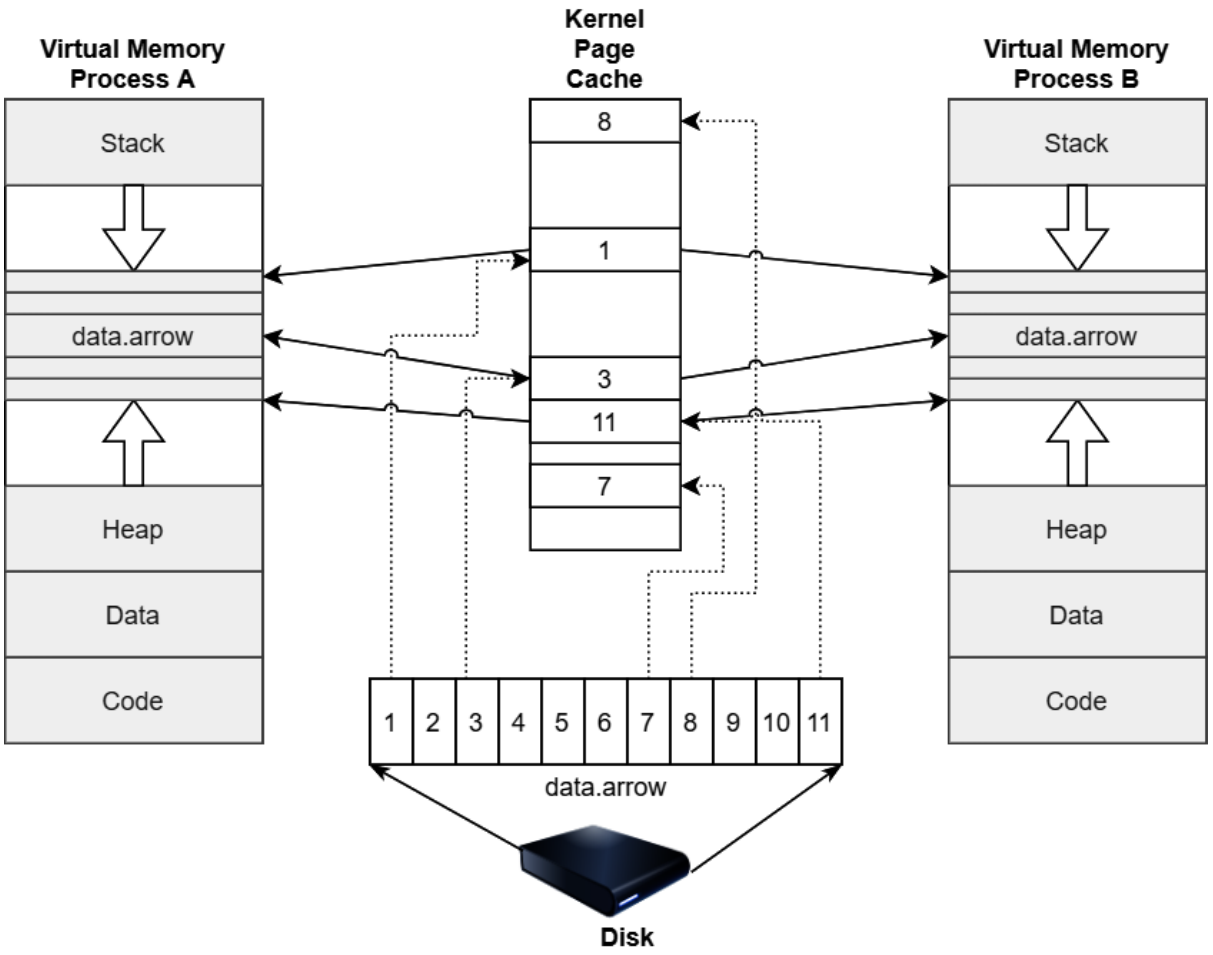
CSV	**1.8 GB**	*calc mean*
	pandas read_csv	20.2 s ± 400ms
	pyarrow.csv.read_csv	816 ms ± 27ms
Parquet	**286 MB**	*calc mean*
	pandas/pyarrow	230 ms

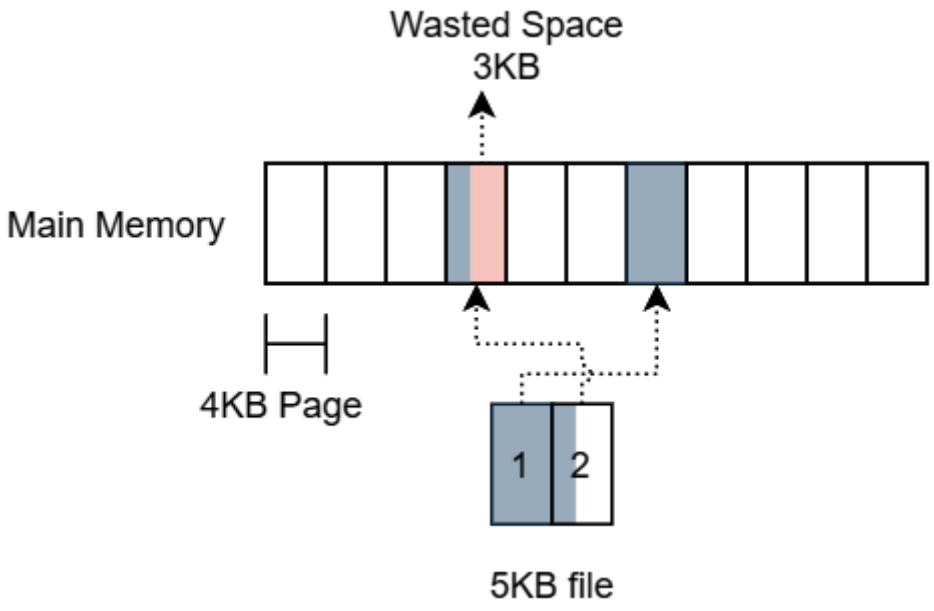
CSV	**1.8 GB**	*calc mean*
	pandas read_csv	20.2 s ± 400ms
	pyarrow.csv.read_csv	816 ms ± 27ms
Parquet	**286 MB**	*calc mean*
	pandas/pyarrow	230 ms
Arrow IPC	**1.77 GB**	*calc mean*
	pyarrow mmap	200 ms

Memory Usage and Runtime




Virtual Memory
(Per Process)
Stack
Heap
Data
Code
Physical RAM
Shared Memory
Some Other Process'
Memory
Disk


Virtual Memory
Process A
Kernel
Page
Cache
Virtual Memory
Process B
Stack
data.arrow
Heap
Data
Code
8
1
3
11
7
1 2 3 4 5 6 7 8 9 10 11
data.arrow
Disk


Wasted Space
3KB
Main Memory
4KB Page
1
2
5KB file

Chapter 5: Crossing the Language Barrier with the Arrow C Data API

Arrow Type	Null	Boolean	Int8 Uint8	Int16 Uint16	Int32 Uint32	Int64 Uint64	Float16
Format String	n	B	c C	s S	i I	l L	e

Arrow Type	Float32	Float64	Binary	Large Binary	UTF-8 String	Large UTF-8 String
Format String	f	g	z	Z	u	U

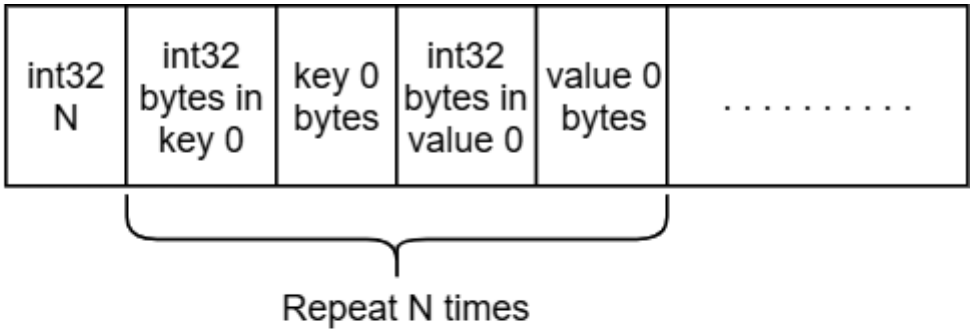
Arrow Type	Format String
Decimal128 [precision 19, scale 10]	d:19,10
Decimal Bit width = NNN [precision 19, scale 10]	d:19,10,NNN
Fixed-Width Binary [42 bytes]	w:42

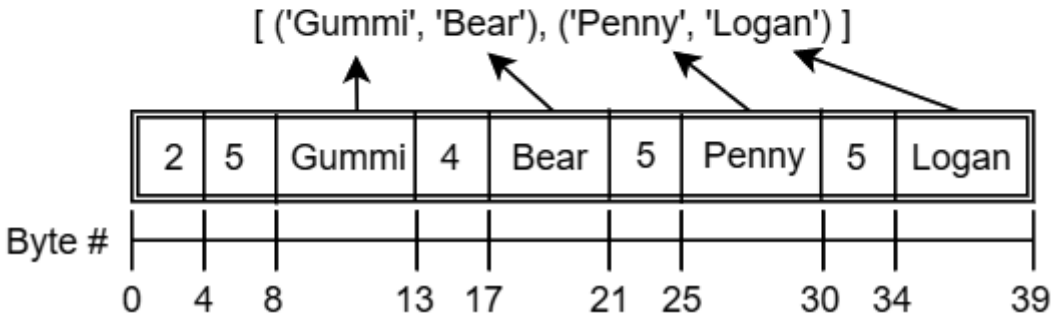
Arrow Type	**Date32** **[days]**	**Date64** **[milliseconds]**	**Time32** **[seconds]** **[milliseconds]**	**Time64** **[microseconds]** **[nanoseconds]**
Format String	tdD	tdm	tts ttm	ttu ttn

Arrow Type	**Duration** **[seconds]** **[milliseconds]**	**Duration** **[microseconds]** **[nanoseconds]**	**Interval** **[months]** **[day, time]**	**Interval** **[month, day, nanoseconds]**
Format String	tDs tDm	tDu tDn	tiM tiD	tin

Arrow Type	**Timestamp with Timezone "..."** **[seconds]** **[milliseconds]**	**Timestamp with Timezone "..."** **[microseconds]** **[nanoseconds]**
Format String	tss:... tsm:...	tsu:... tsn:...

Arrow Type	**List**	**Large List**	**Fixed-size List [123 items]**	**Struct**	**Map**	**Dense Union type-ids I,J,...**	**Sparse Union type-ids I,J,...**
Format String	+l	+L	+w:123	+s	+m	+ud:I,J,...	+us:I,J,...



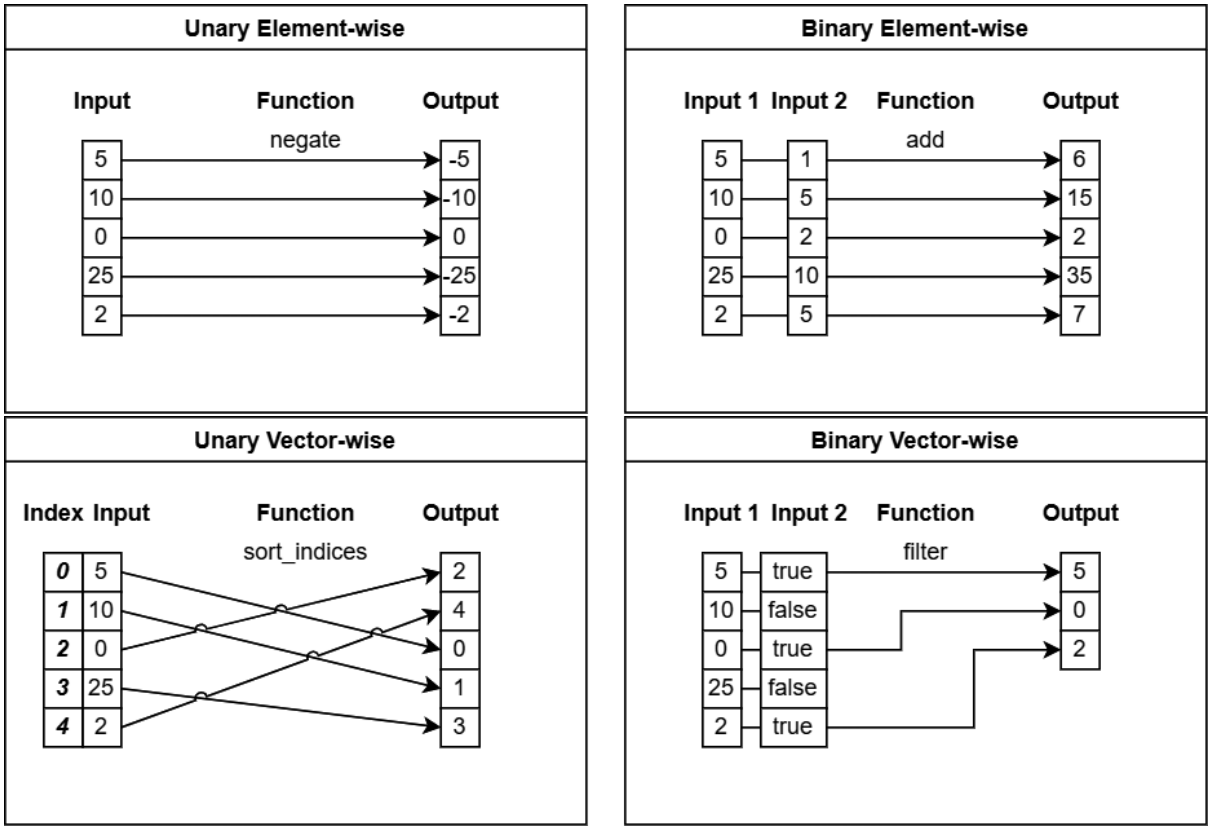


Scalar Aggregations

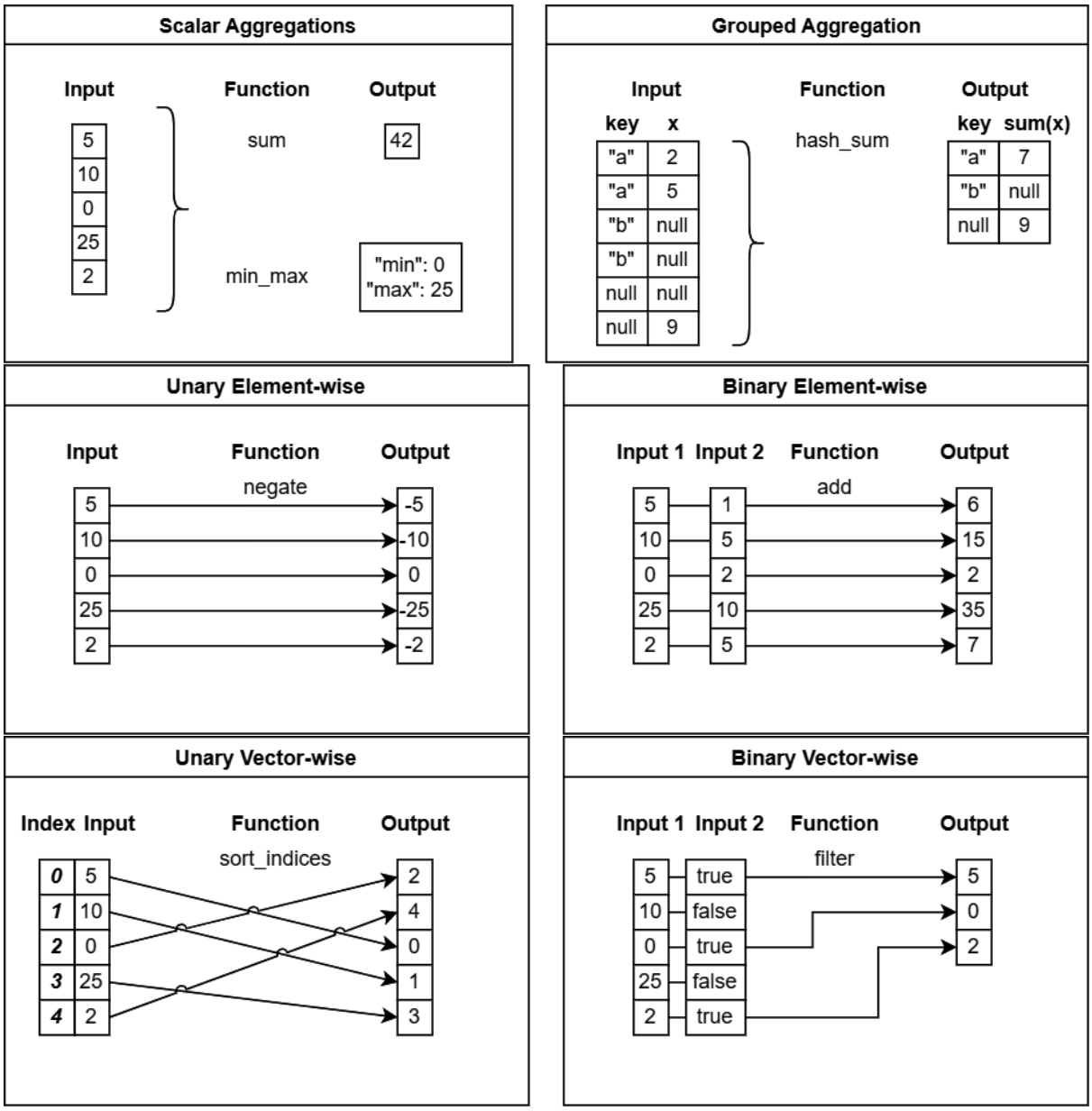
Input	Function	Output
5 10 0 25 2	sum	42
	min_max	"min": 0 "max": 25

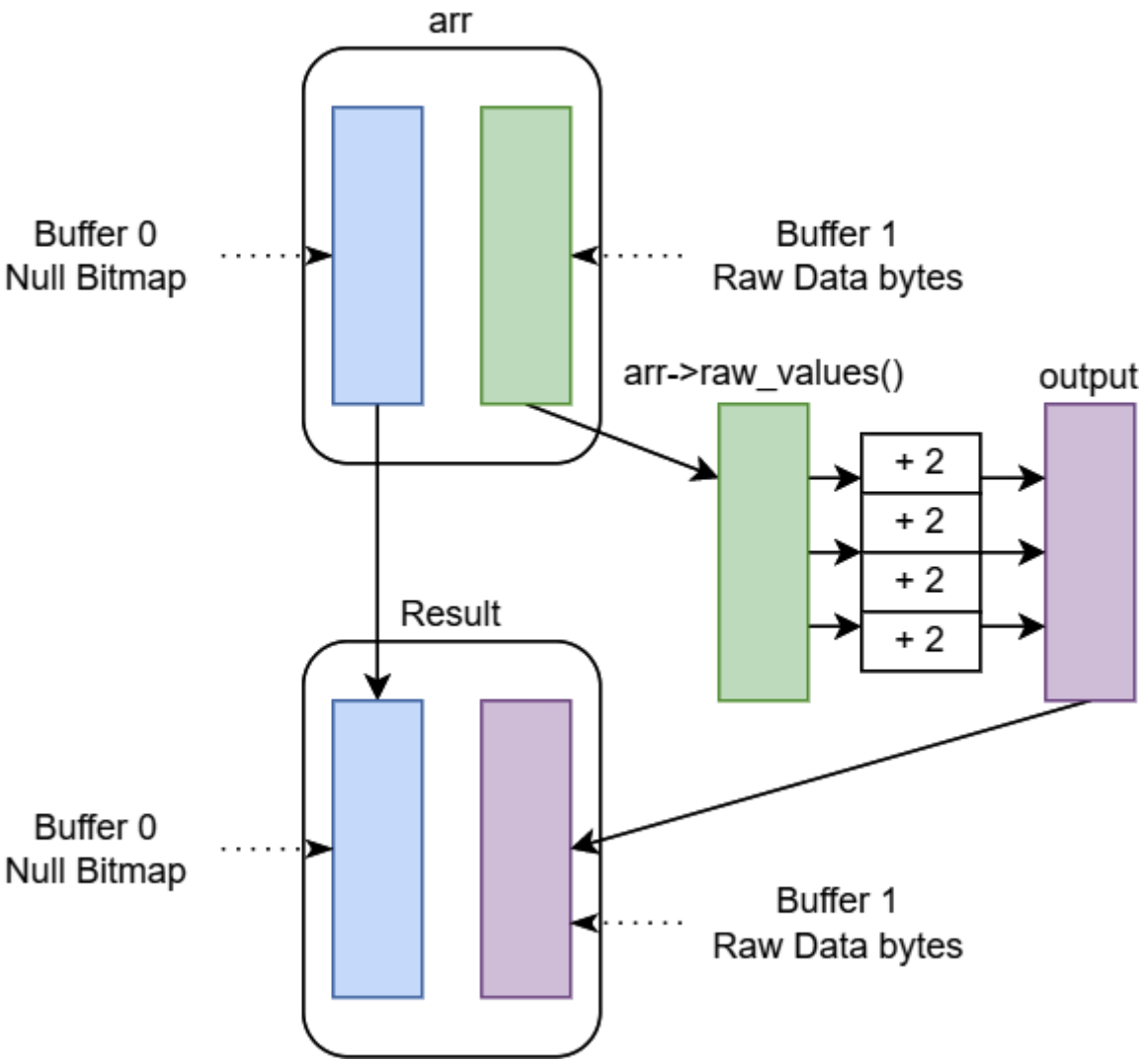
Grouped Aggregation

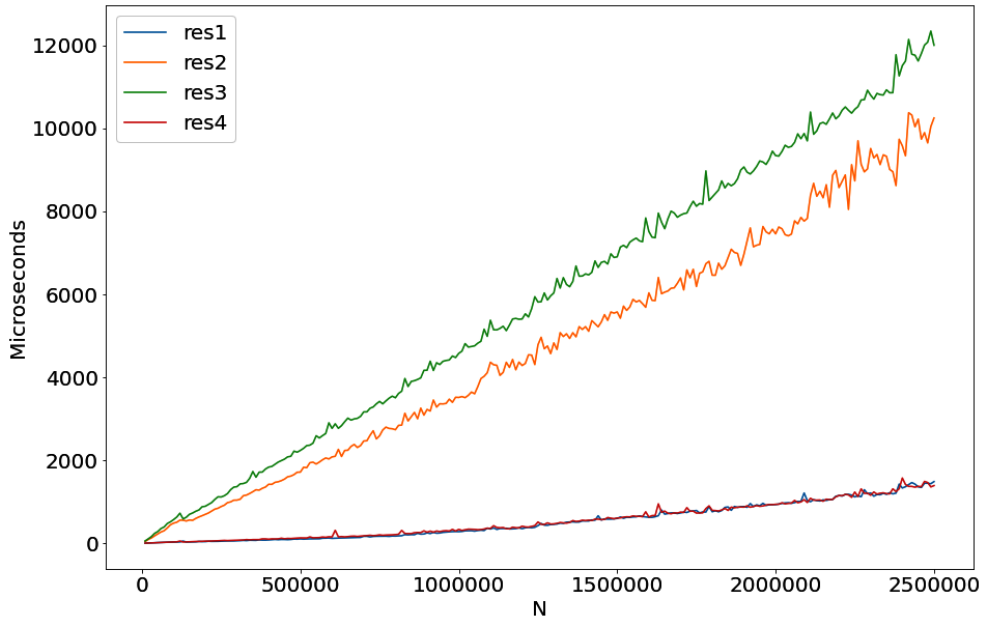
Input		Function	Output	
key	x		key	sum(x)
"a"	2	hash_sum	"a"	7
"a"	5		"b"	null
"b"	null		null	9
"b"	null			
null	null			
null	9			


Unary Element-wise
Input
Function
Output
negate
5
10
0
25
2
-5
-10
0
-25
-2
Binary Element-wise
Input 1
Input 2
Function
Output
add
5
10
0
25
2
1
5
2
10
5
6
15
2
35
7
Unary Vector-wise
Index
Input
Function
Output
sort_indices
0
1
2
3
4
5
10
0
25
2
2
4
0
1
3
Binary Vector-wise
Input 1
Input 2
Function
Output
filter
5
10
0
25
2
true
false
true
false
true
5
0
2

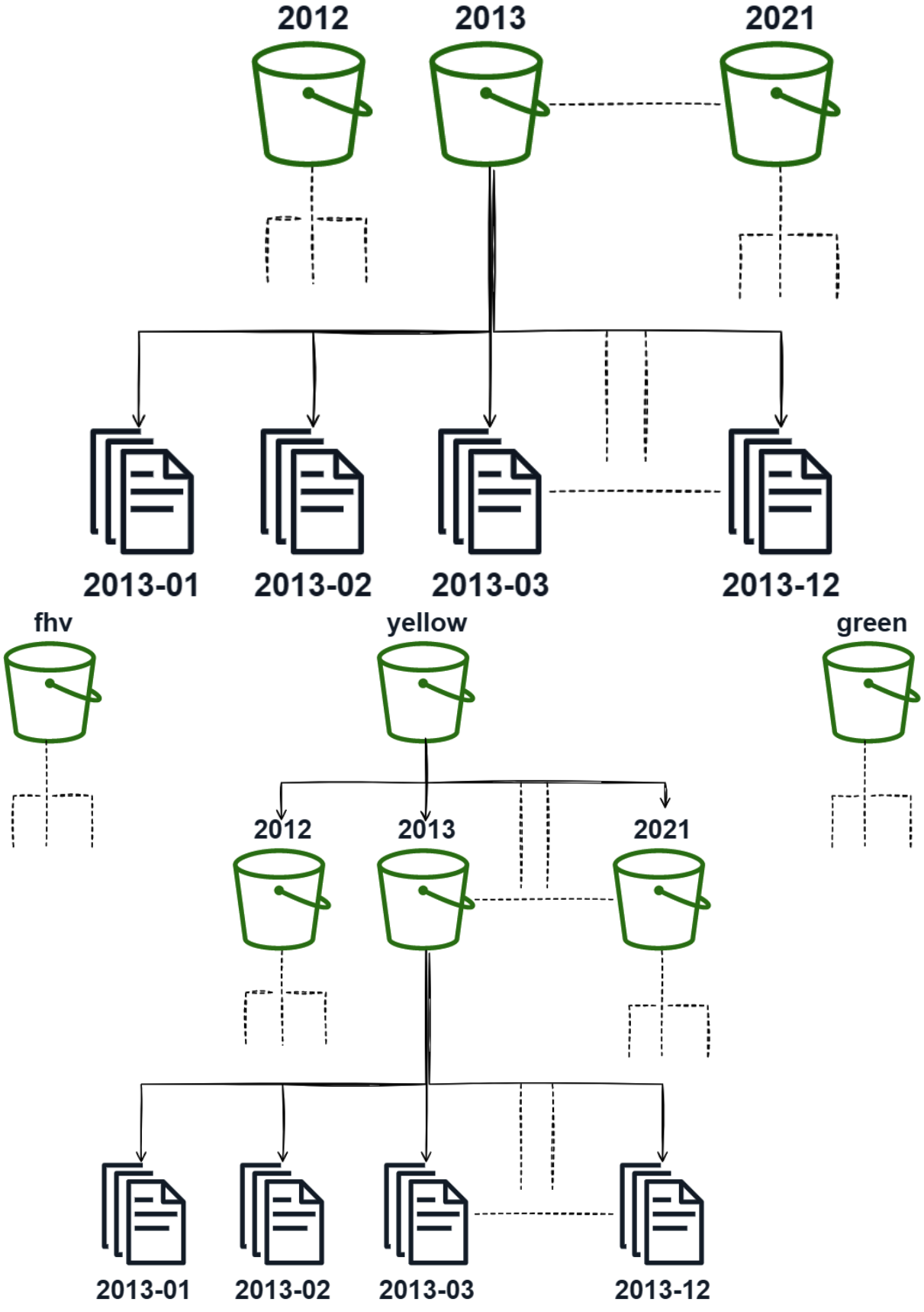
Chapter 6: Leveraging the Arrow Compute APIs




arr
Buffer 0
Null Bitmap
Buffer 1
Raw Data bytes
arr->raw_values()
output
+ 2
+ 2
+ 2
+ 2
Result
Buffer 0
Null Bitmap
Buffer 1
Raw Data bytes


res1
res2
res3
res4
12000
10000
8000
6000
4000
2000
0
Microseconds
0
500000
1000000
1500000
2000000
2500000
N

Chapter 7: Using the Arrow Datasets API



data1.parquet

a	b	c
0	9	1
1	8	2
2	7	1
3	6	2
4	5	1

data2.parquet

a	b	c
5	4	2
6	3	1
7	2	2
8	1	1
9	0	2

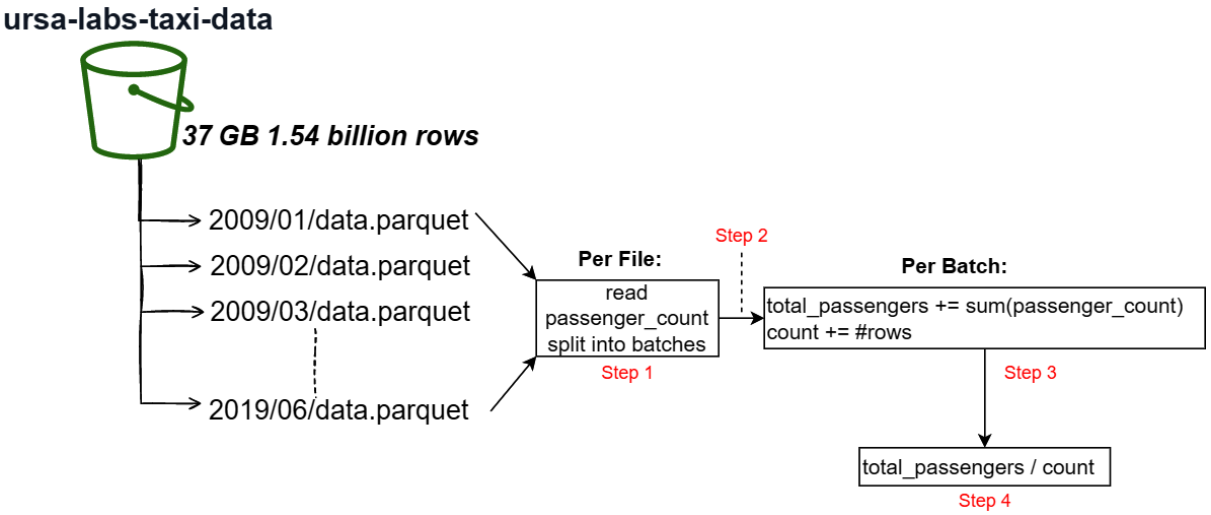
Input **Expression** **Output**

a	b	c
0	9	1
1	8	2
2	7	1
3	6	2
4	5	1

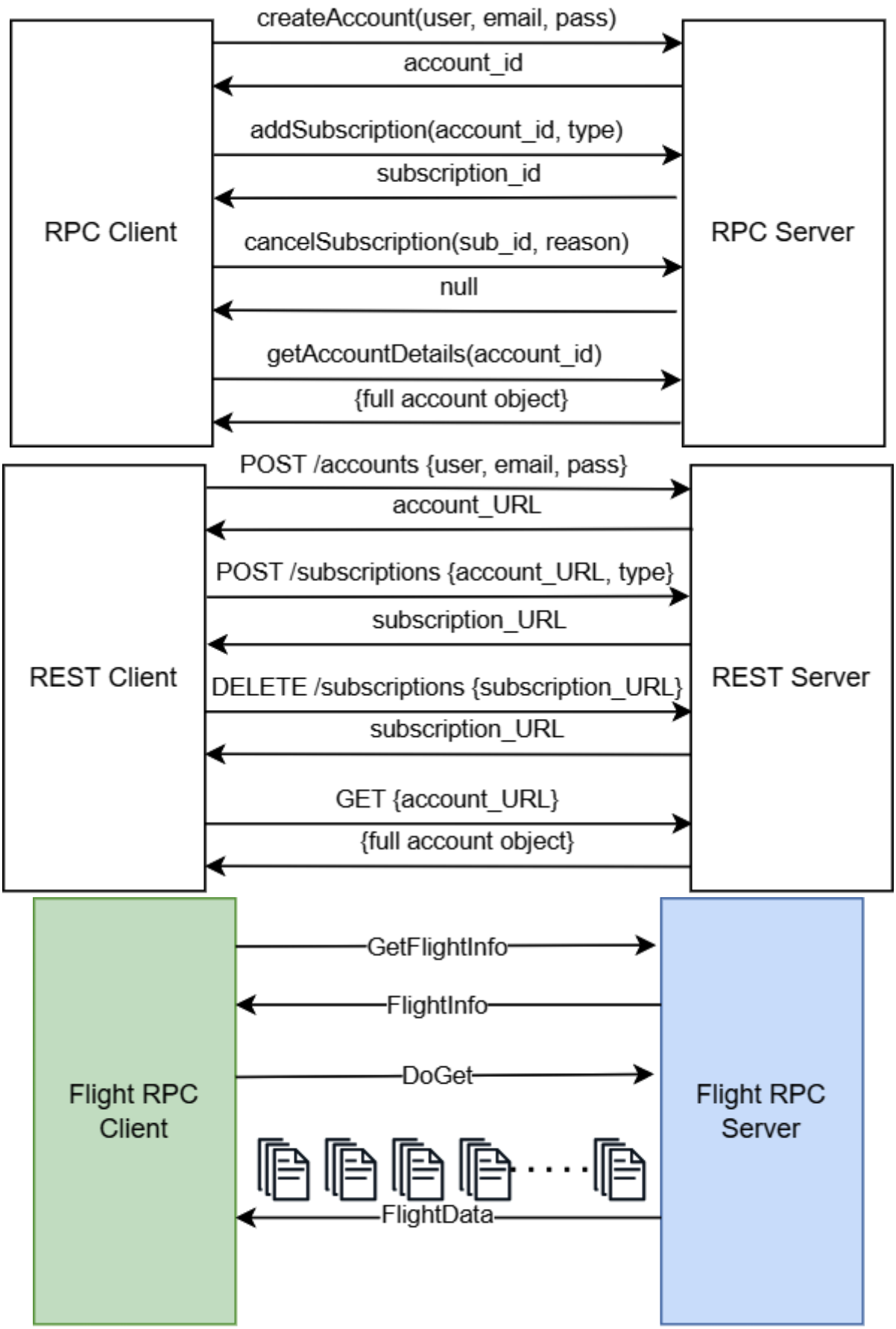
field_ref("b")

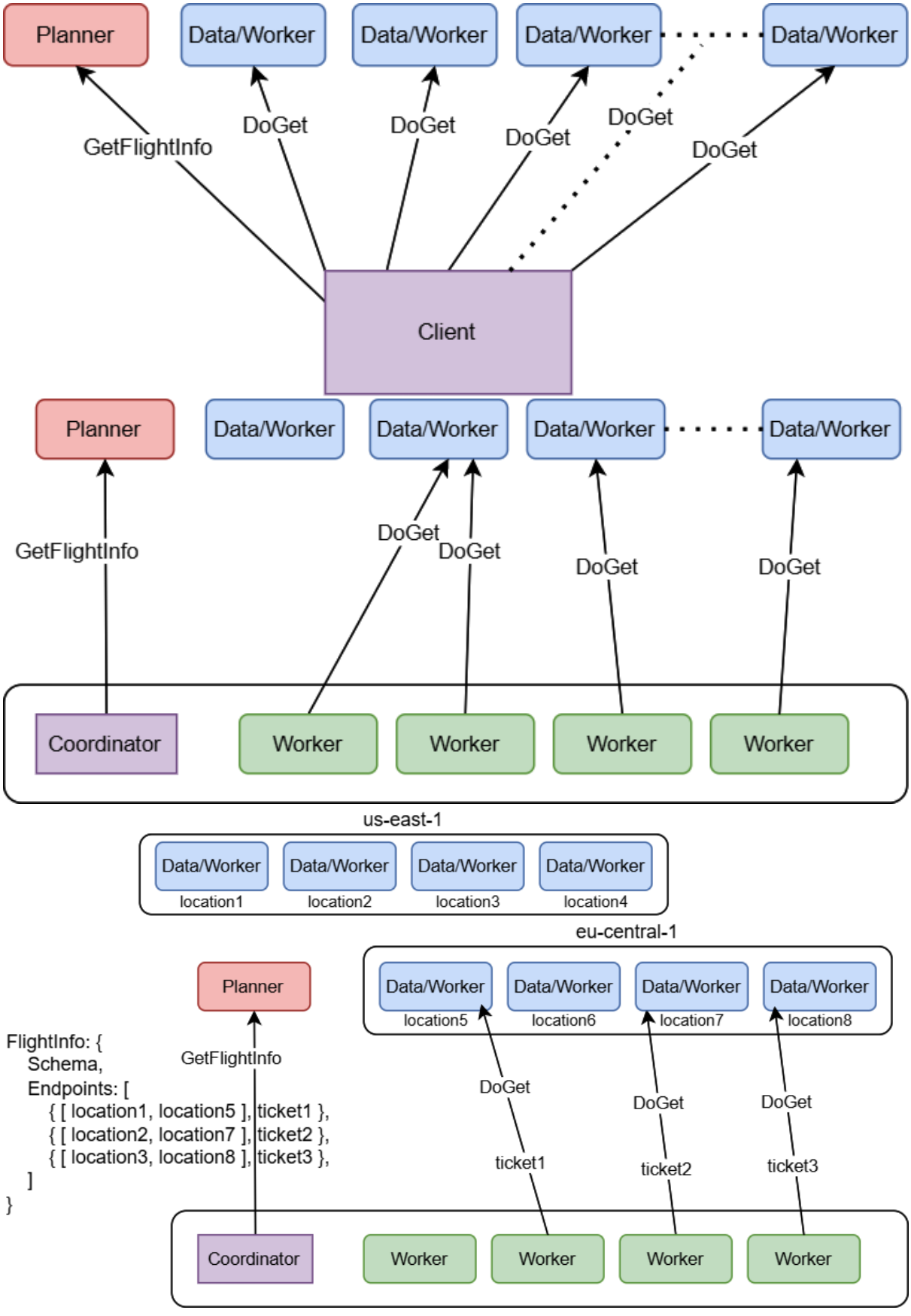
9
8
7
6
5

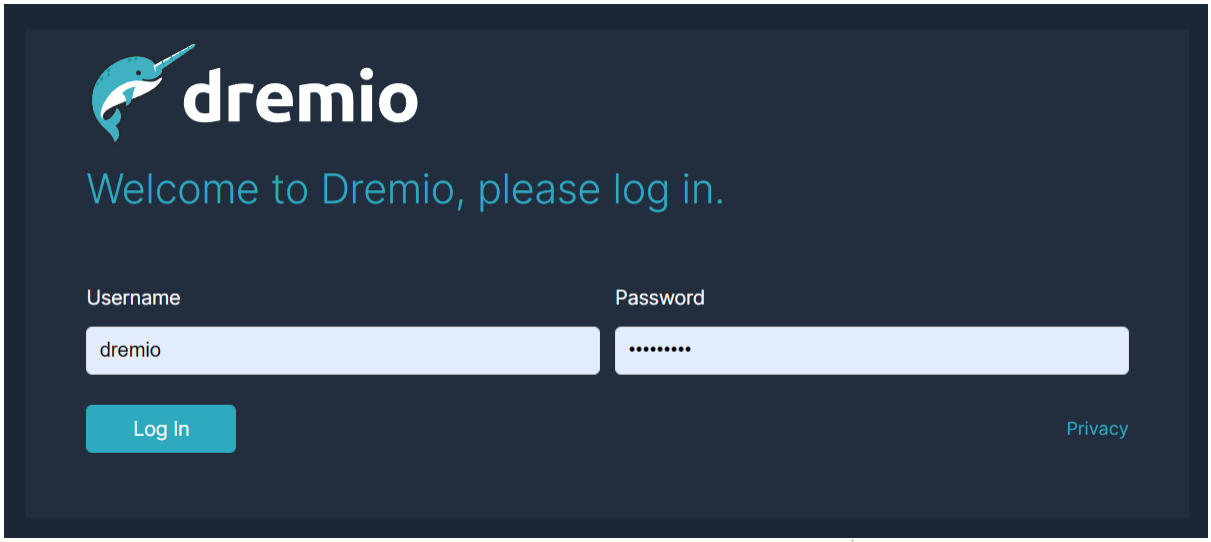
Input			Expression	Output
a	b	c		
0	9	1	call("less",	true
1	8	2	[field_ref("a"), literal(4)])	true
2	7	1		true
3	6	2		true
4	5	1		false

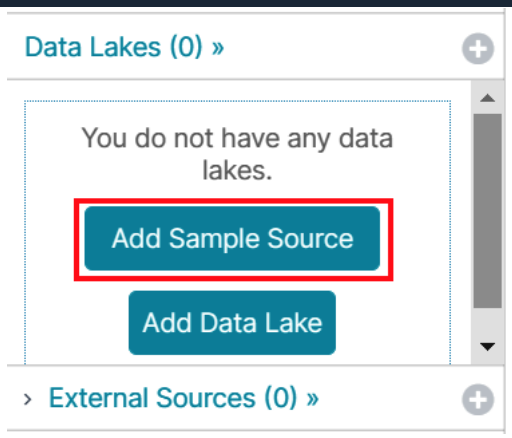


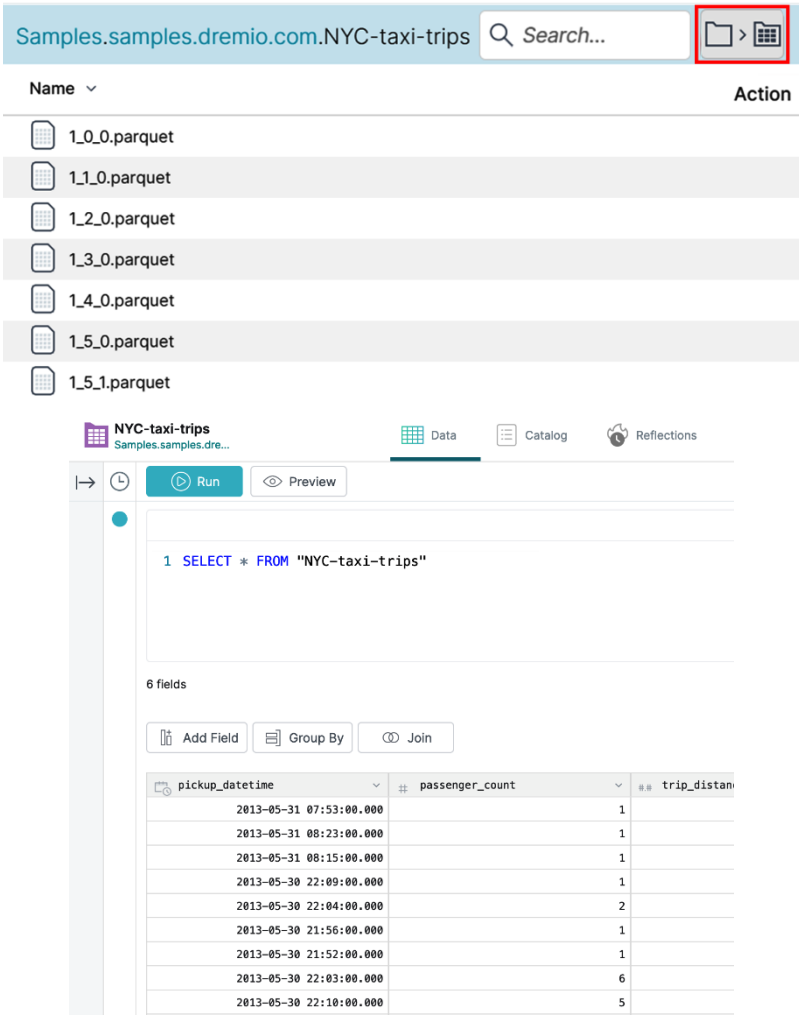
Chapter 8: Exploring Apache Arrow Flight RPC



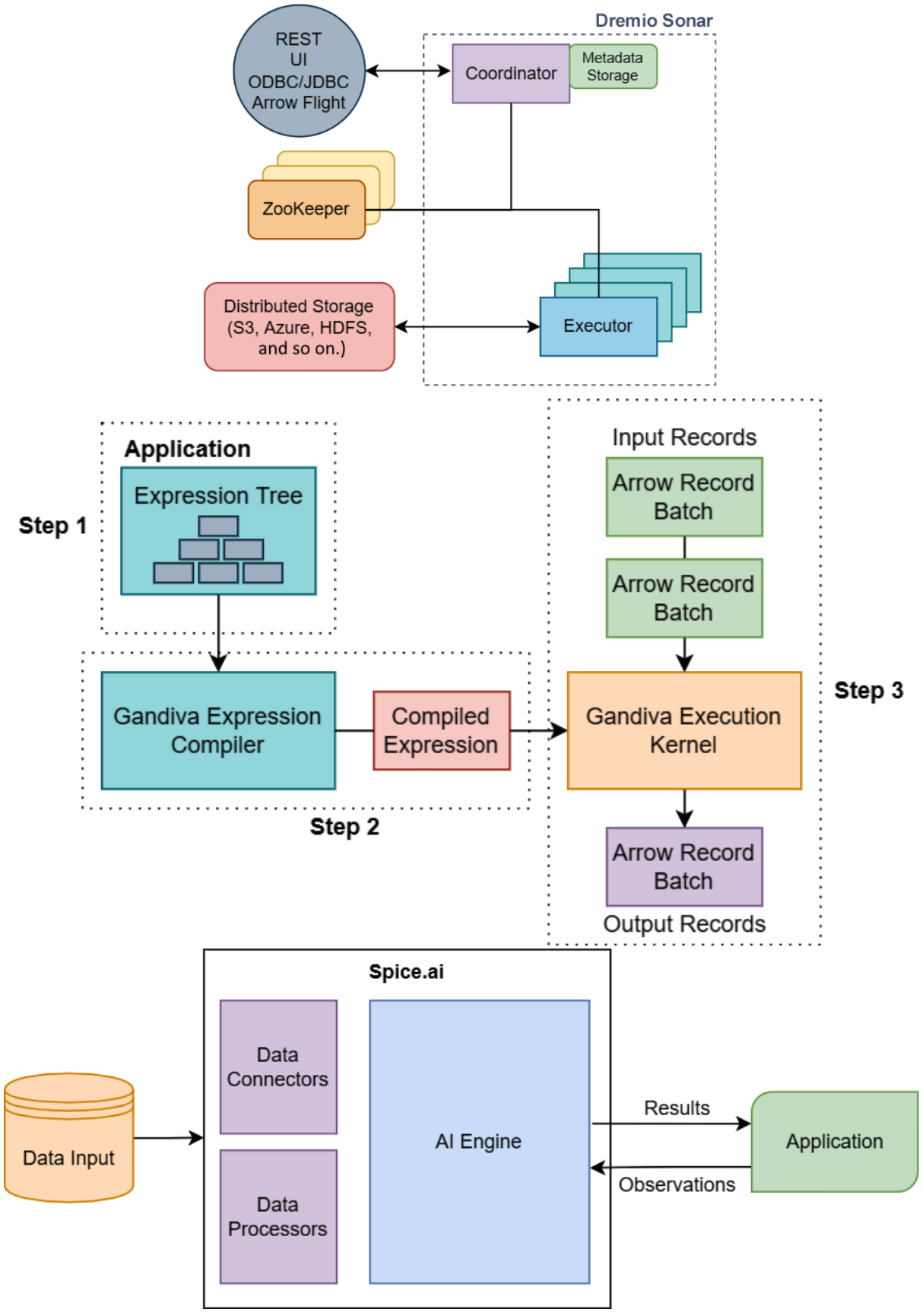

Planner
Data/Worker
Data/Worker
Data/Worker
Data/Worker
GetFlightInfo
DoGet
DoGet
DoGet
DoGet
DoGet
Client
Planner
Data/Worker
Data/Worker
Data/Worker
Data/Worker
GetFlightInfo
DoGet
DoGet
DoGet
DoGet
Coordinator
Worker
Worker
Worker
Worker
us-east-1
Data/Worker
Data/Worker
Data/Worker
Data/Worker
location1
location2
location3
location4
eu-central-1
Planner
Data/Worker
Data/Worker
Data/Worker
Data/Worker
location5
location6
location7
location8
FlightInfo: {
Schema,
Endpoints: [
{ [location1, location5], ticket1 },
{ [location2, location7], ticket2 },
{ [location3, location8], ticket3 },
]
}
GetFlightInfo
DoGet
DoGet
DoGet
ticket1
ticket2
ticket3
Coordinator
Worker
Worker
Worker
Worker


dremio
Welcome to Dremio, please log in.
Username
dremio
Password
Log In
Privacy

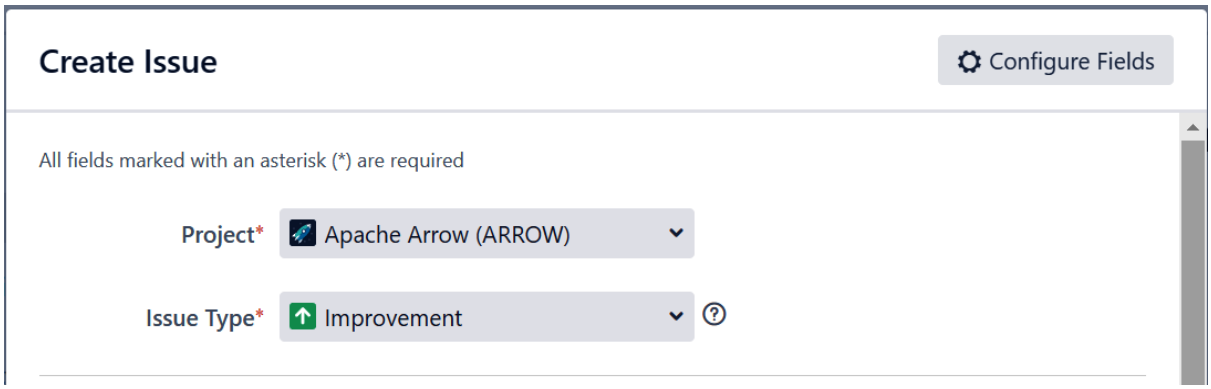

Data Lakes (0) »
You do not have any data lakes.
Add Sample Source
Add Data Lake
External Sources (0) »

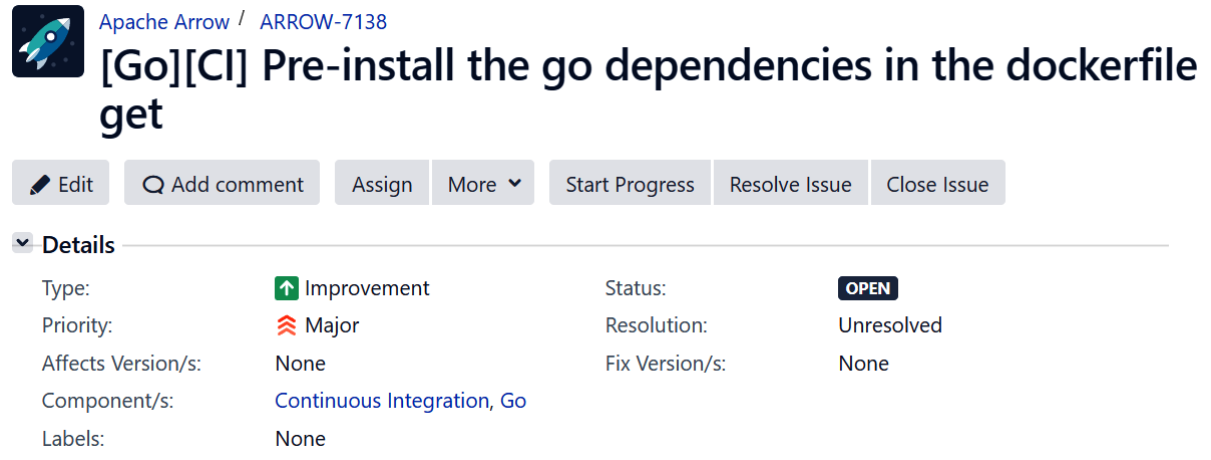

Samples.samples.dremio.com.NYC-taxi-trips
Search...
Name
Action
1_0_0.parquet
1_1_0.parquet
1_2_0.parquet
1_3_0.parquet
1_4_0.parquet
1_5_0.parquet
1_5_1.parquet
NYC-taxi-trips
Samples.samples.dre...
Data
Catalog
Reflections
Run
Preview
1 SELECT * FROM "NYC-taxi-trips"
6 fields
Add Field
Group By
Join
pickup_datetime
passenger_count
2013-05-31 07:53:00.000 1
2013-05-31 08:23:00.000 1
2013-05-31 08:15:00.000 1
2013-05-30 22:09:00.000 1
2013-05-30 22:04:00.000 2
2013-05-30 21:56:00.000 1
2013-05-30 21:52:00.000 1
2013-05-30 22:03:00.000 6
2013-05-30 22:10:00.000 5

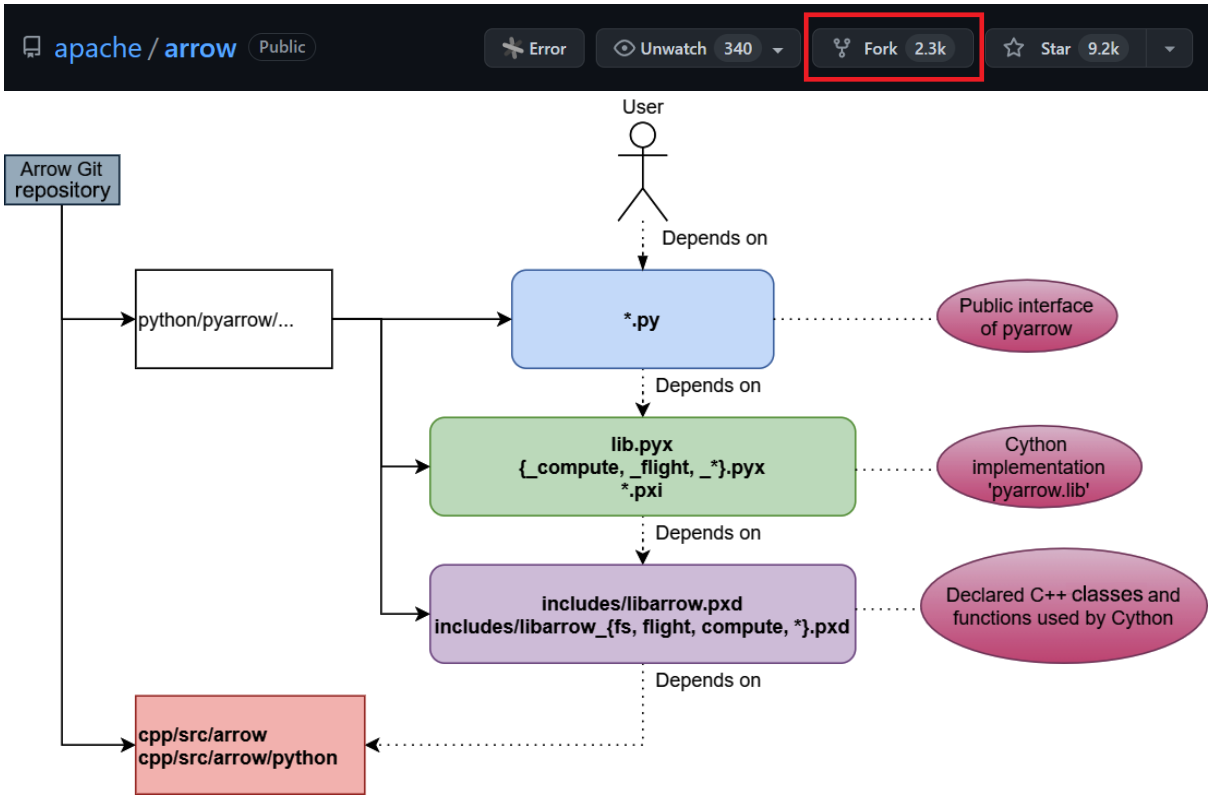
Chapter 9: Powered By Apache Arrow

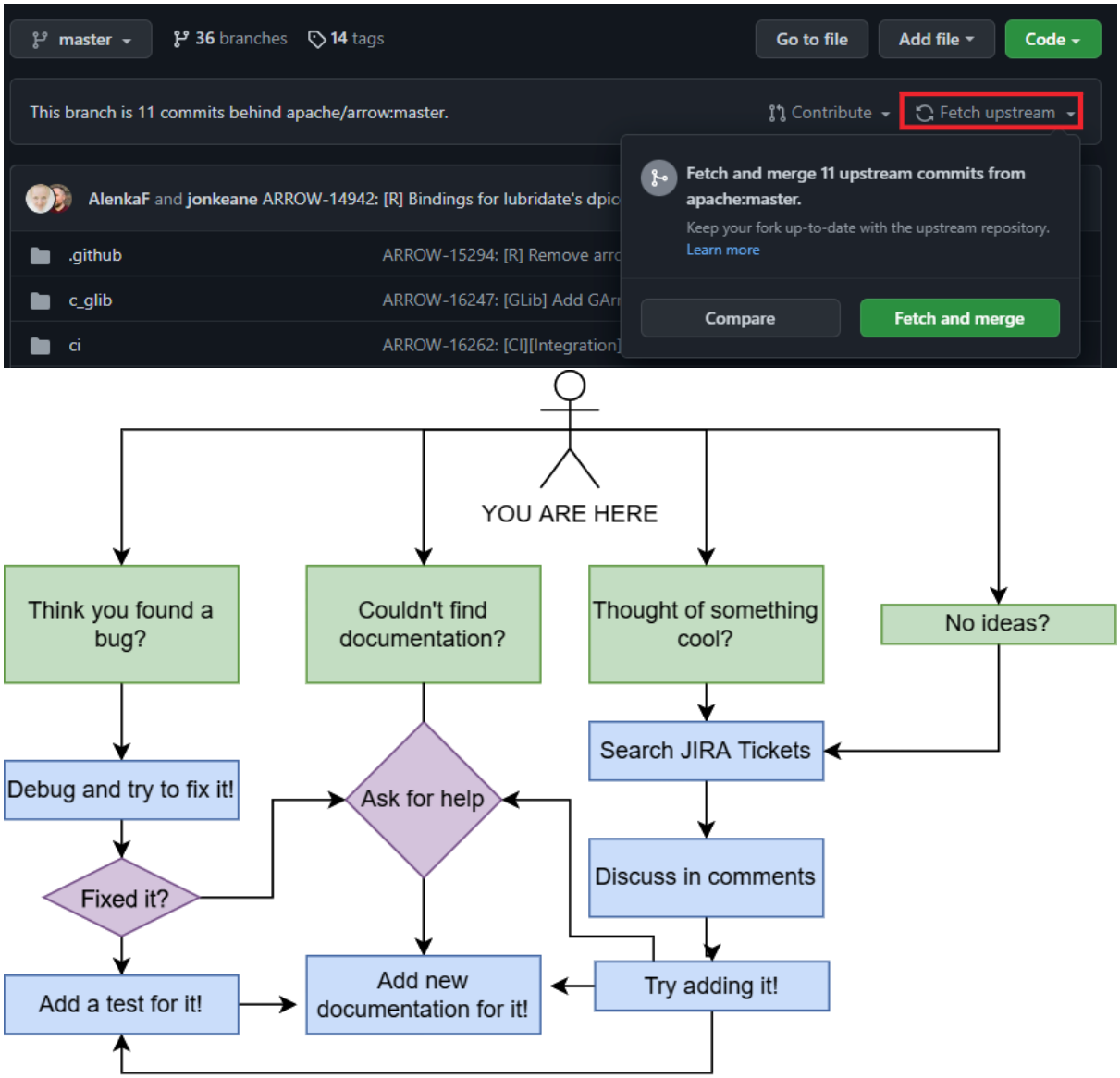


Chapter 10: How to Leave Your Mark on Arrow

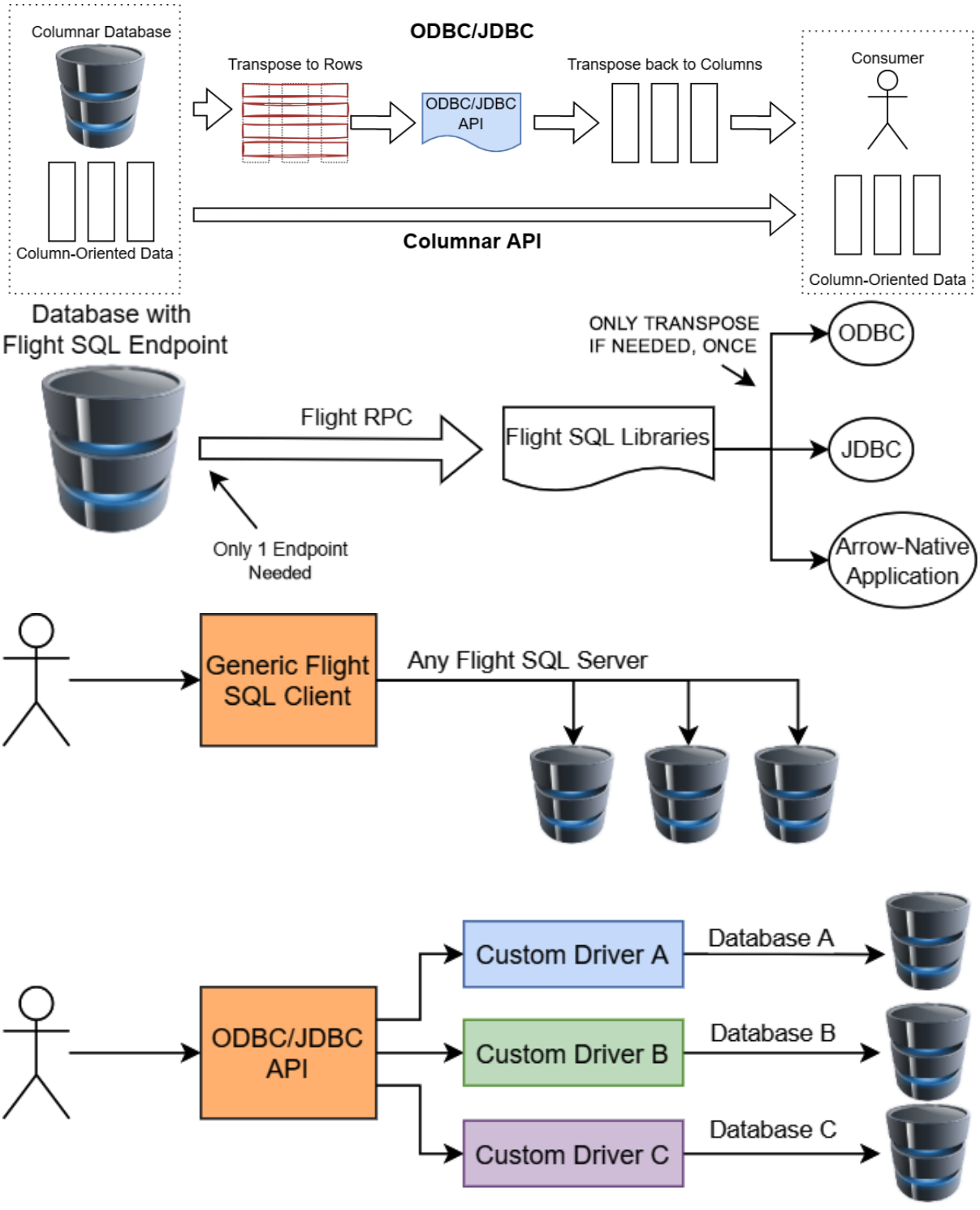


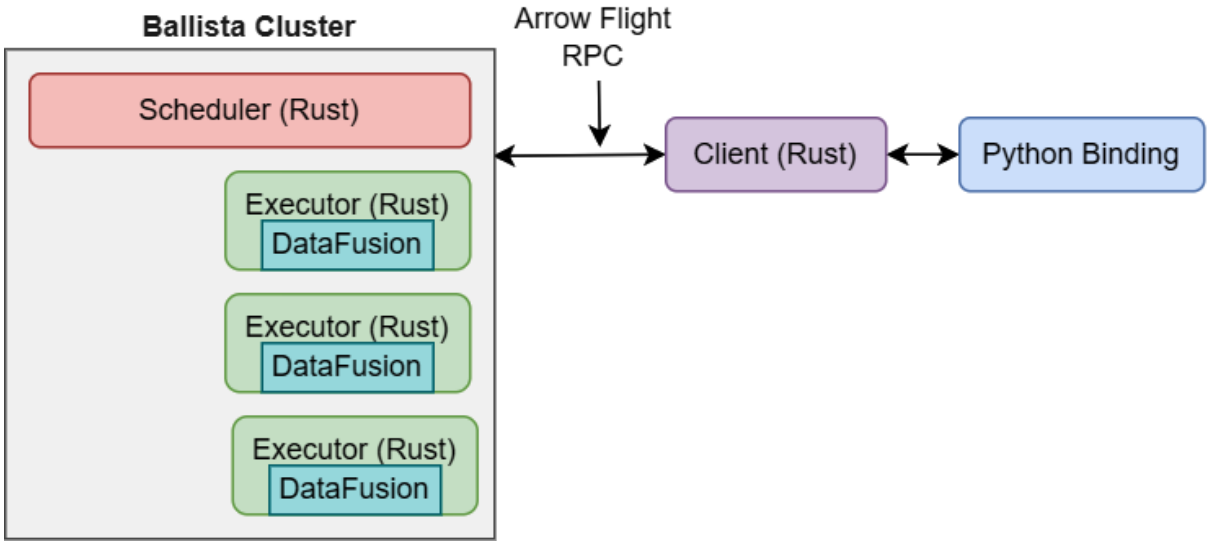


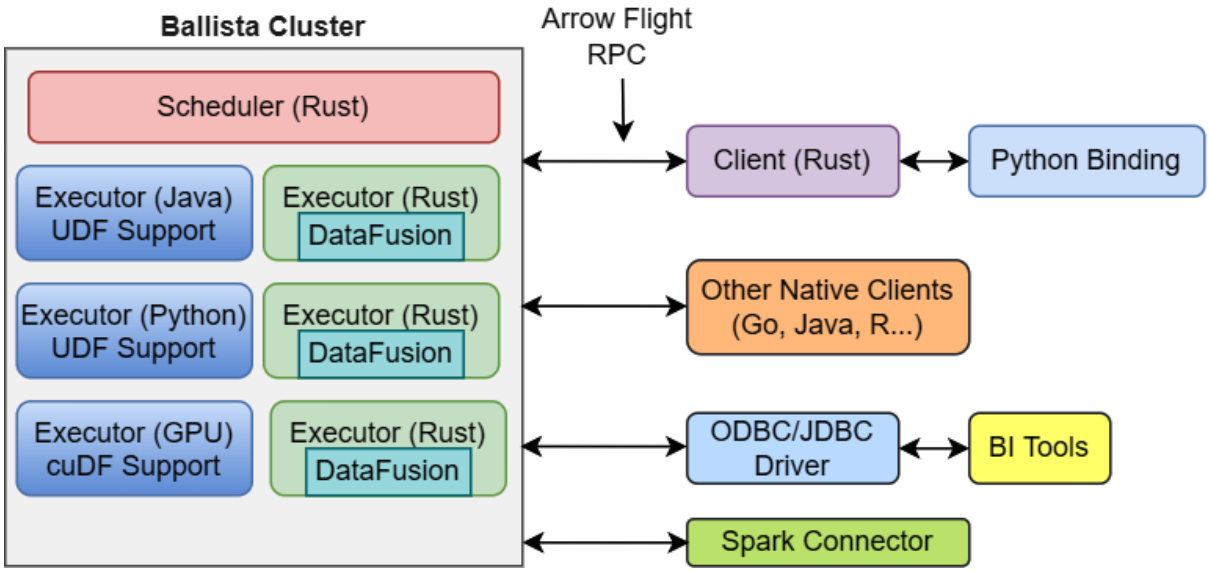


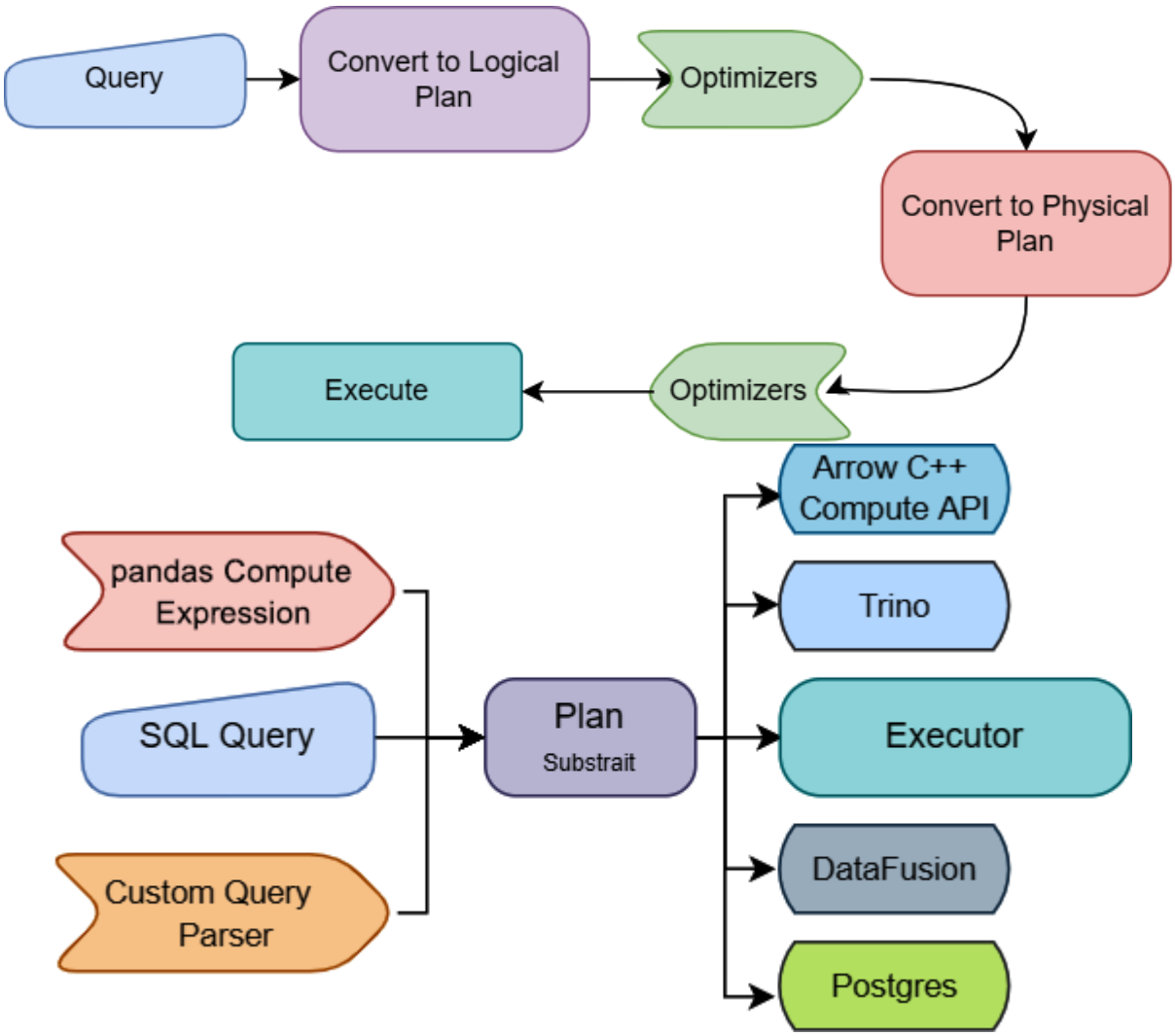

master
36 branches
14 tags
Go to file
Add file
Code
This branch is 11 commits behind apache/arrow:master.
Contribute
Fetch upstream
Fetch and merge 11 upstream commits from apache:master.
Keep your fork up-to-date with the upstream repository.
Learn more
Compare
Fetch and merge
AlenkaF and jonkeane ARROW-14942: [R] Bindings for lubridate's dpic
.github
ARROW-15294: [R] Remove arr
c_glib
ARROW-16247: [GLib] Add GAr
ci
ARROW-16262: [CI][Integration]
YOU ARE HERE
Think you found a bug?
Couldn't find documentation?
Thought of something cool?
No ideas?
Debug and try to fix it!
Ask for help
Search JIRA Tickets
Fixed it?
Discuss in comments
Add a test for it!
Add new documentation for it!
Try adding it!

Chapter 11: Future Development and Plans




Ballista Cluster
Scheduler (Rust)
Executor (Rust)
DataFusion
Executor (Rust)
DataFusion
Executor (Rust)
DataFusion
Arrow Flight
RPC
Client (Rust)
Python Binding


Ballista Cluster
Scheduler (Rust)
Executor (Java)
UDF Support
Executor (Rust)
DataFusion
Executor (Python)
UDF Support
Executor (Rust)
DataFusion
Executor (GPU)
cuDF Support
Executor (Rust)
DataFusion
Arrow Flight
RPC
Client (Rust)
Python Binding
Other Native Clients
(Go, Java, R...)
ODBC/JDBC
Driver
BI Tools
Spark Connector


Query
Convert to Logical Plan
Optimizers
Convert to Physical Plan
Optimizers
Execute
pandas Compute Expression
SQL Query
Custom Query Parser
Plan
Substrait
Arrow C++ Compute API
Trino
Executor
DataFusion
Postgres