The International Workshop on Testing Database Systems

dbtest.io

fuzzydata

A Scalable Workload Generator for Testing Dataframe Workflow Systems

Mohammed Suhail Rehman, Aaron Elmore
University of Chicago

Database Benchmarking

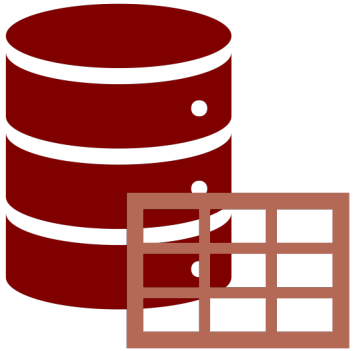
Relational Systems

Wisconsin Benchmark
[BITT'83]
TPC [1988]
...

Key-Value Stores

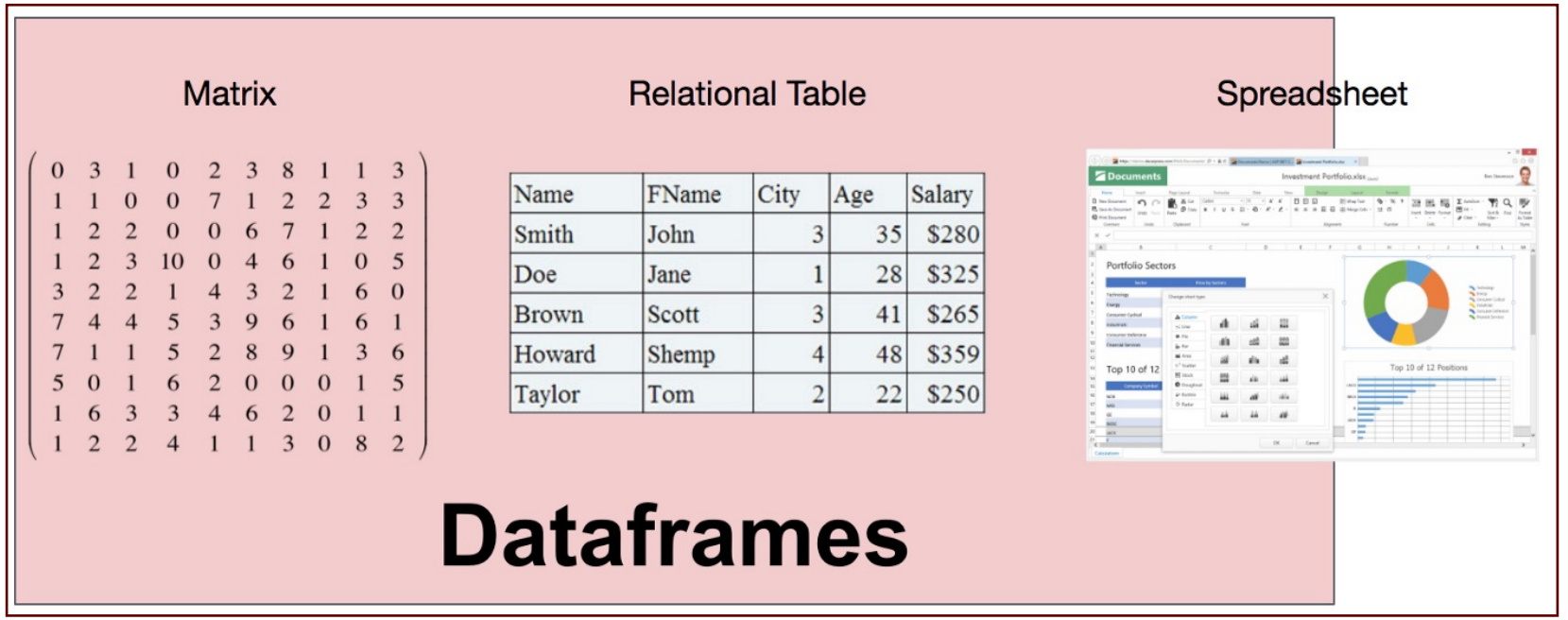
YCSB [Cooper2010]

DataFrame Systems



???

What Makes DataFrames Different?



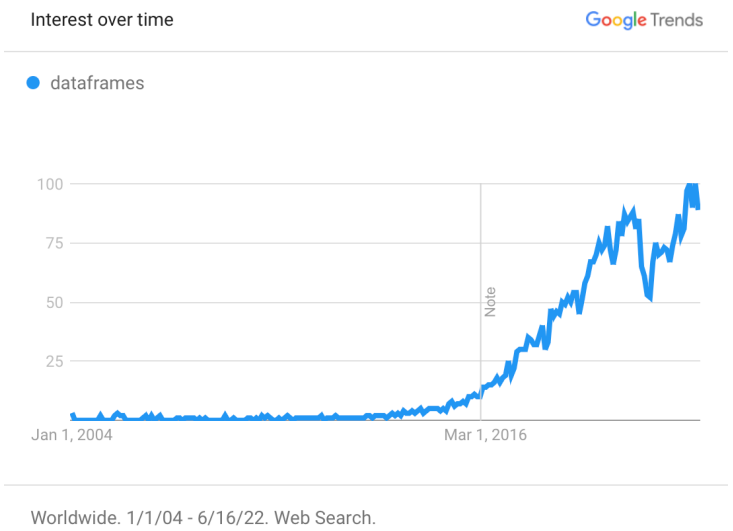
0	3	1	0	2	3	8	1	1	3
1	1	0	0	7	1	2	2	3	3
1	2	2	0	0	6	7	1	2	2
1	2	3	10	0	4	6	1	0	5
3	2	2	1	4	3	2	1	6	0
7	4	4	5	3	9	6	1	6	1
7	1	1	5	2	8	9	1	3	6
5	0	1	6	2	0	0	0	1	5
1	6	3	3	4	6	2	0	1	1
1	2	2	4	1	1	3	0	8	2

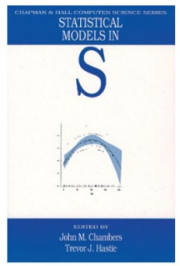
Name	FName	City	Age	Salary
Smith	John	3	35	$280
Doe	Jane	1	28	$325
Brown	Scott	3	41	$265
Howard	Shemp	4	48	$359
Taylor	Tom	2	22	$250

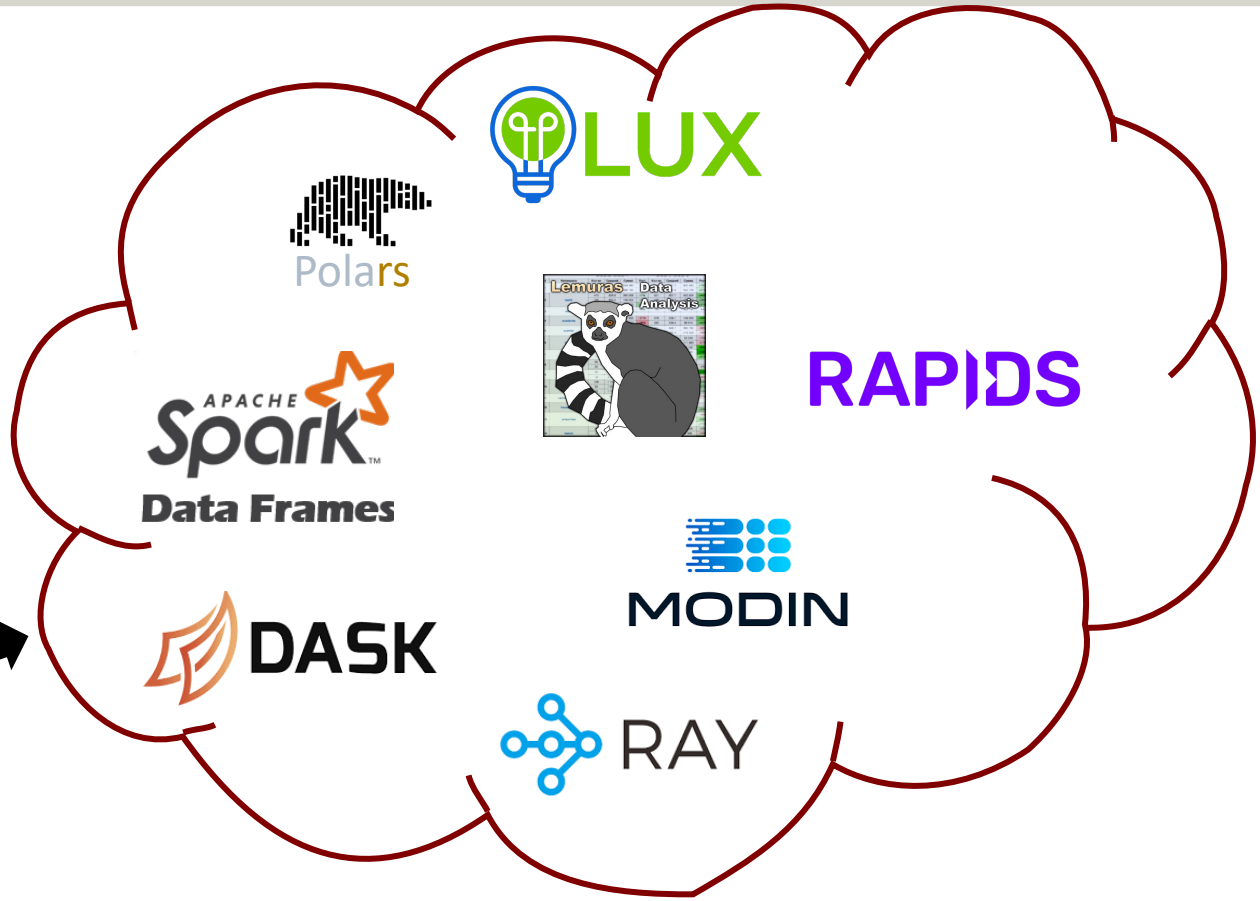
Dataframes allow for a mix of **linear-algebra**, **relational** and **spreadsheet** operations on tabular data using a programmatic API.

Images From Devin Petershon (https://towardsdatascience.com/preventing-the-death-of-the-dataframe-8bca1c0f83c8)

History and Usage of Dataframes







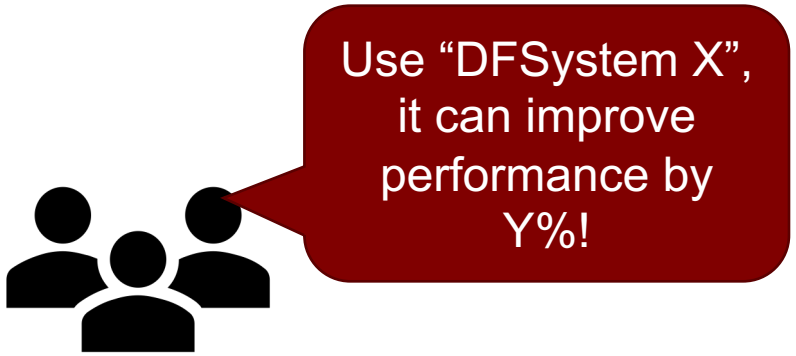
S [1976]
Similar to Relational Tables but Matrix-like

R [1993]
Open-Sourced in R

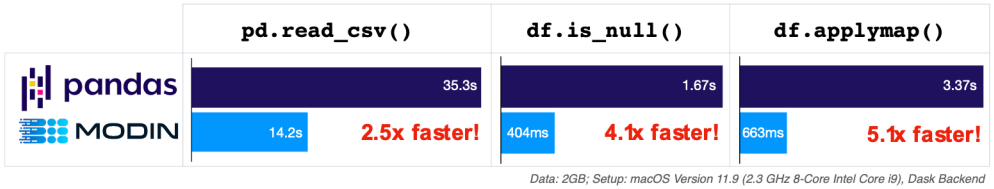
Pandas [2008]
Brought to Python by Pandas

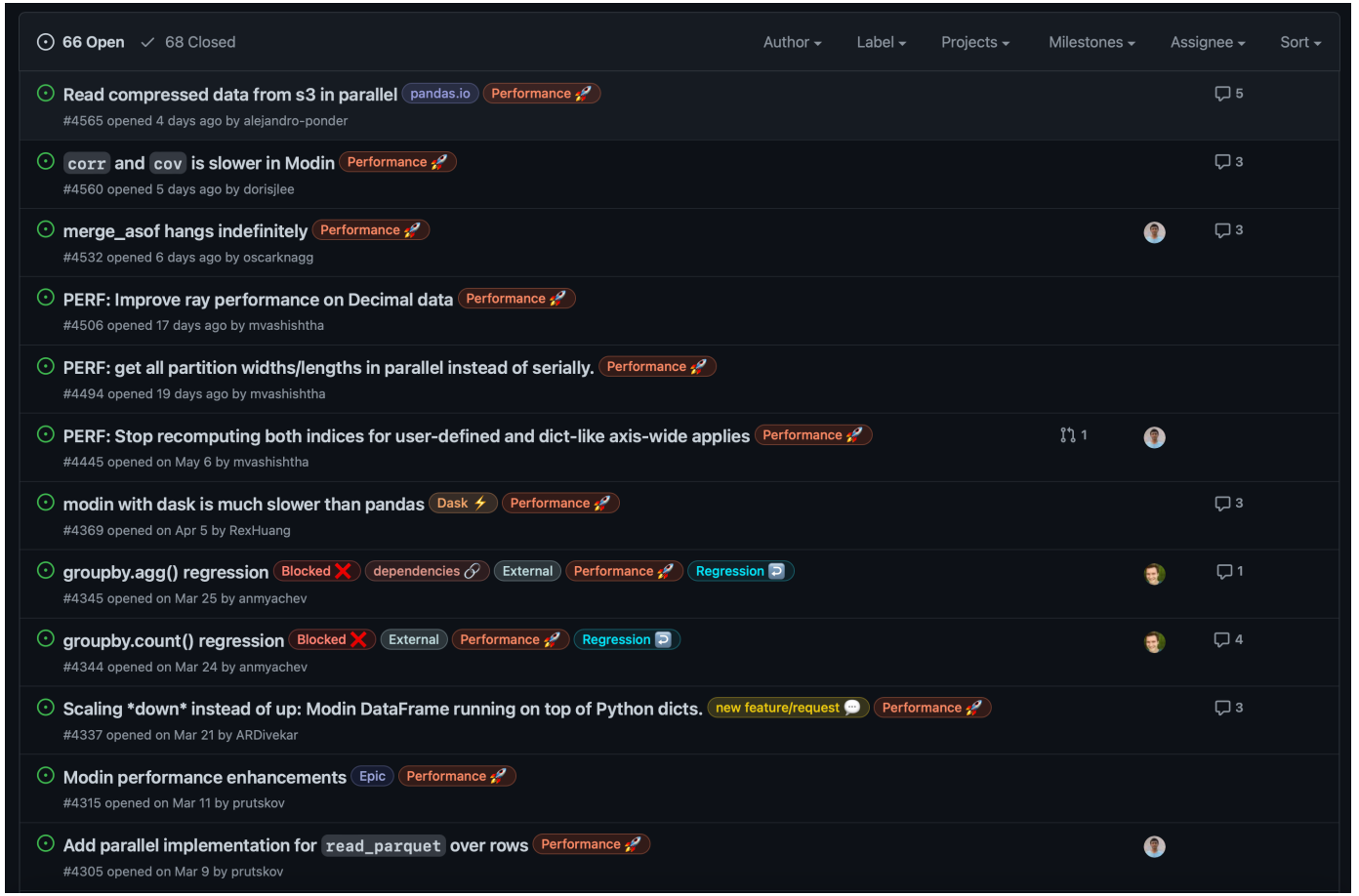
Images From Devin Petershon (https://towardsdatascience.com/preventing-the-death-of-the-dataframe-8bca1c0f83c8)

Performance Evaluation for DataFrame Systems



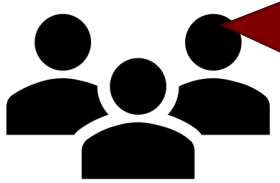
(Mostly) Anecdotal Performance Reports





Users face performance issues in practice.

Performance Evaluation for DataFrame Systems

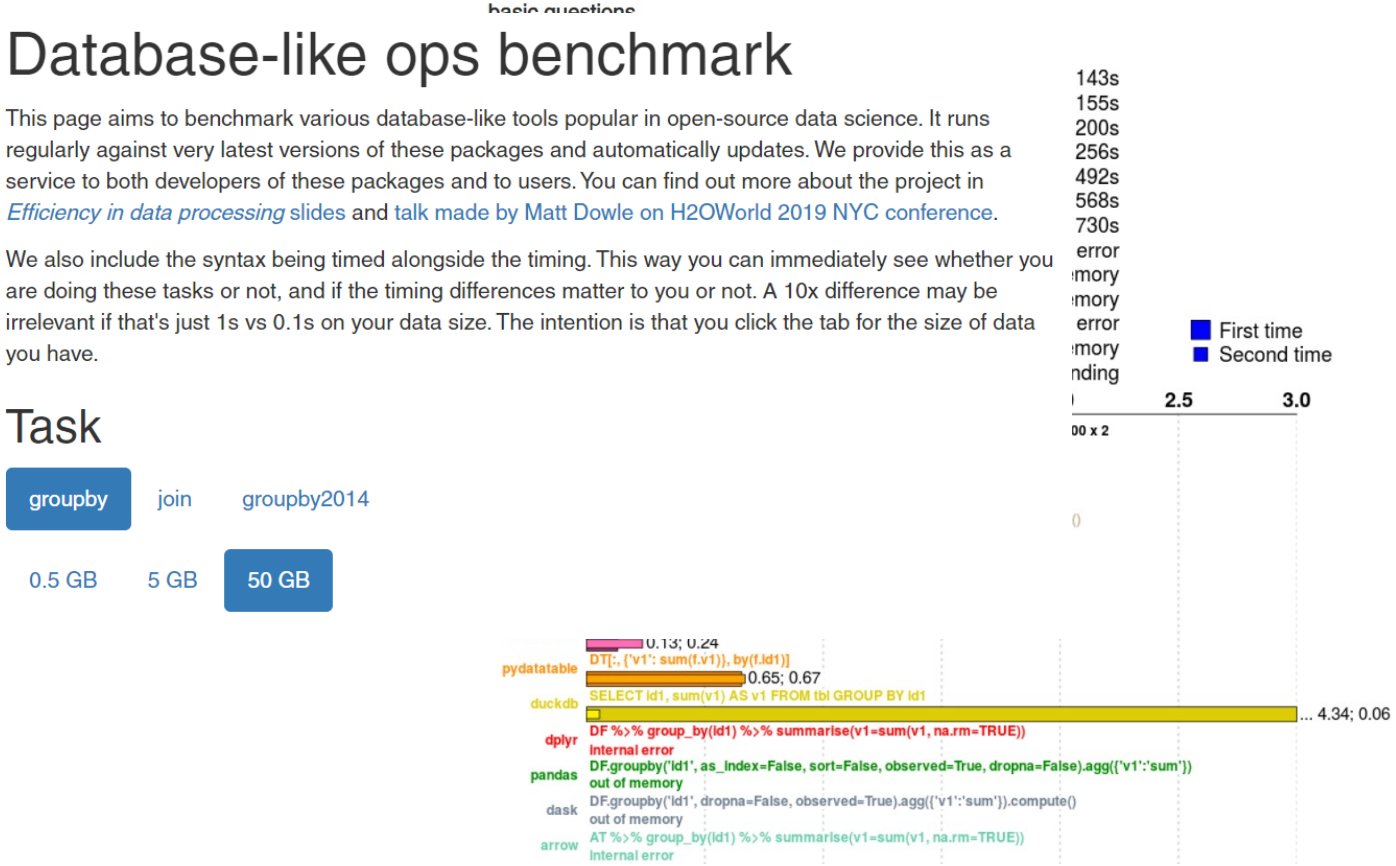


Use “DFSystem X”, it can improve performance by Y%!

(Mostly) Anecdotal Performance Reports

Homogenous Data Types Evaluation of only Relational Operations

Current “Benchmarks” have Limited Coverage



Need a Robust, Reproduceable Workload Generation System tailored for the DataFrame API.

Workload Generation Requirements

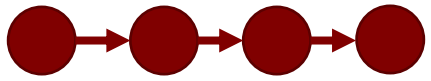
- Realistic Artifacts – Allow for different data types, entropies in column values etc.
- Generate workflows with different structures – linear to star-like to mimic different data analysis patterns

In [33]: `data`

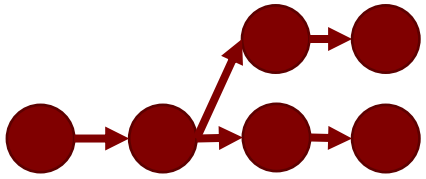
Out[33]:

	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude	longitude	...	Y2004	Y2005	Y2006	Y2007	Y2008	Y2009
0	AF	2	Afghanistan	2511	Wheat and products	5142	Food	1000 tonnes	33.94	67.71	...	3249.0	3486.0	3704.0	4164.0	4252.0	4538.0
1	AF	2	Afghanistan	2805	Rice (Milled Equivalent)	5142	Food	1000 tonnes	33.94	67.71	...	419.0	445.0	546.0	455.0	490.0	415.0
2	AF	2	Afghanistan	2513	Barley and products	5521	Feed	1000 tonnes	33.94	67.71	...	58.0	236.0	262.0	263.0	230.0	379.0
3	AF	2	Afghanistan	2513	Barley and products	5142	Food	1000 tonnes	33.94	67.71	...	185.0	43.0	44.0	48.0	62.0	55.0
4	AF	2	Afghanistan	2514	Maize and products	5521	Feed	1000 tonnes	33.94	67.71	...	120.0	208.0	233.0	249.0	247.0	195.0
5	AF	2	Afghanistan	2514	Maize and products	5142	Food	1000 tonnes	33.94	67.71	...	231.0	67.0	82.0	67.0	69.0	71.0
6	AF	2	Afghanistan	2517	Millet and products	5142	Food	1000 tonnes	33.94	67.71	...	15.0	21.0	11.0	19.0	21.0	18.0
7	AF	2	Afghanistan	2520	Cereals, Other	5142	Food	1000 tonnes	33.94	67.71	...	2.0	1.0	1.0	0.0	0.0	0.0
8	AF	2	Afghanistan	2531	Potatoes and products	5142	Food	1000 tonnes	33.94	67.71	...	276.0	294.0	294.0	260.0	242.0	250.0
9	AF	2	Afghanistan	2536	Sugar cane	5521	Feed	1000 tonnes	33.94	67.71	...	50.0	29.0	61.0	65.0	54.0	114.0
10	AF	2	Afghanistan	2537	Sugar beet	5521	Feed	1000 tonnes	33.94	67.71	...	0.0	0.0	0.0	0.0	0.0	0.0

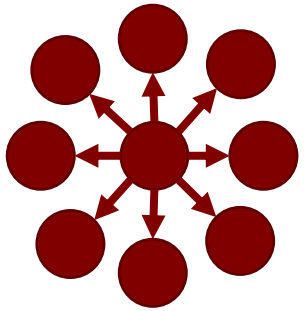
Realistic Data Values



Linear Workflows



Branching

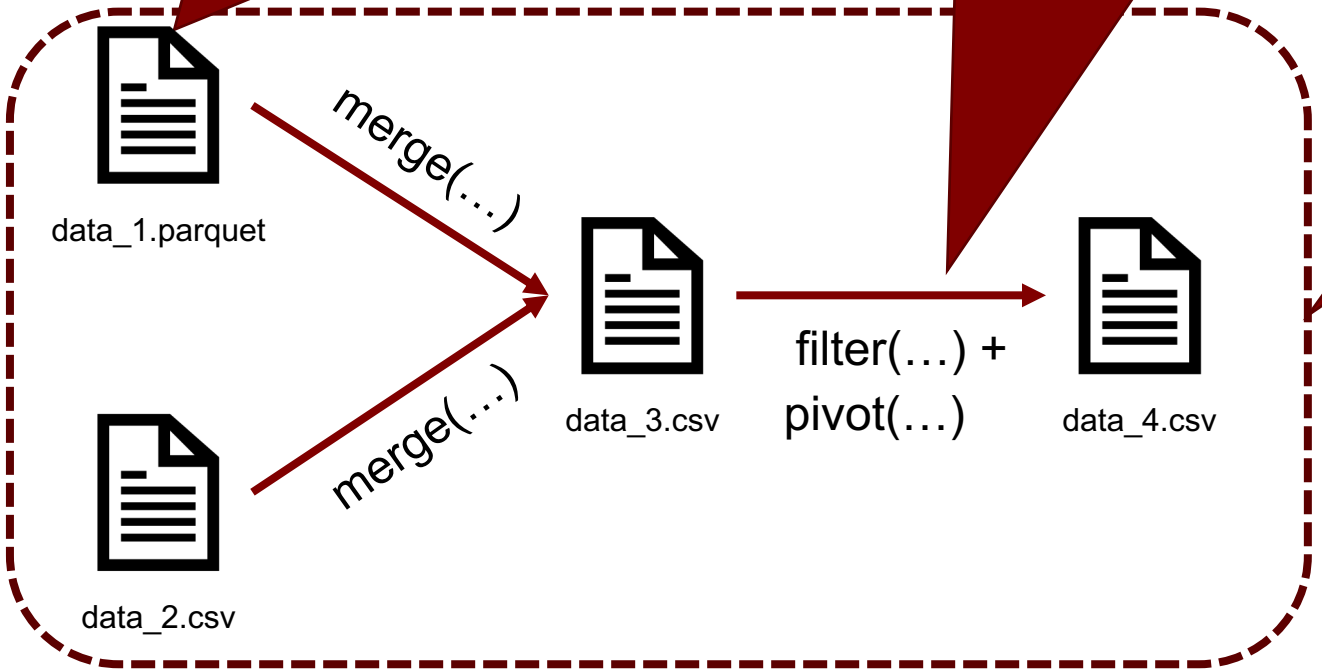


Star-Like

FuzzyData Abstractions

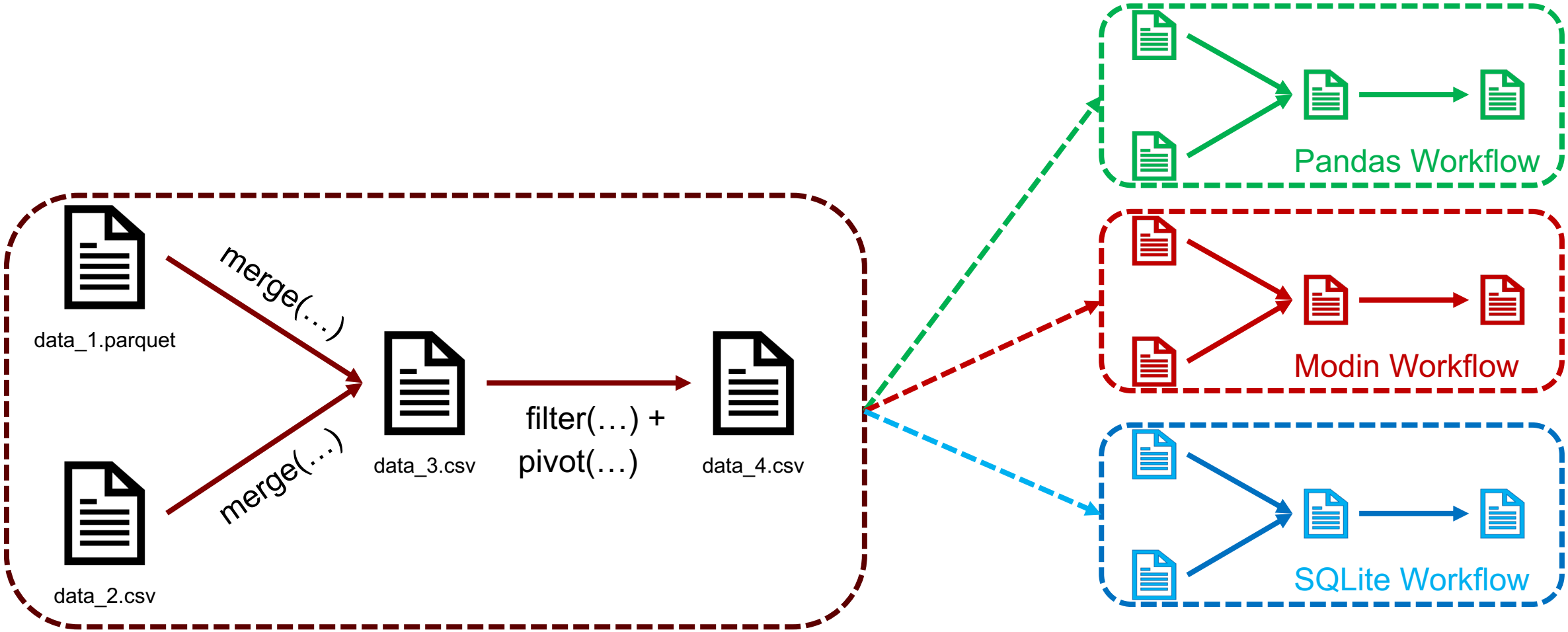
Data Artifacts
- Abstract Representation of A Table
- Table Metadata/Schema
- (de)-Serialization Routines

Operation
- Abstract Representation of Operation(s)
- Argument Specification
- Expected Change in Schema after Operation



Workflow
- List of Operations and Source Artifacts
- Enough information to allow for scaling and replay of the workflow across clients.

FuzzyData Implementations



Random Artifact Generator

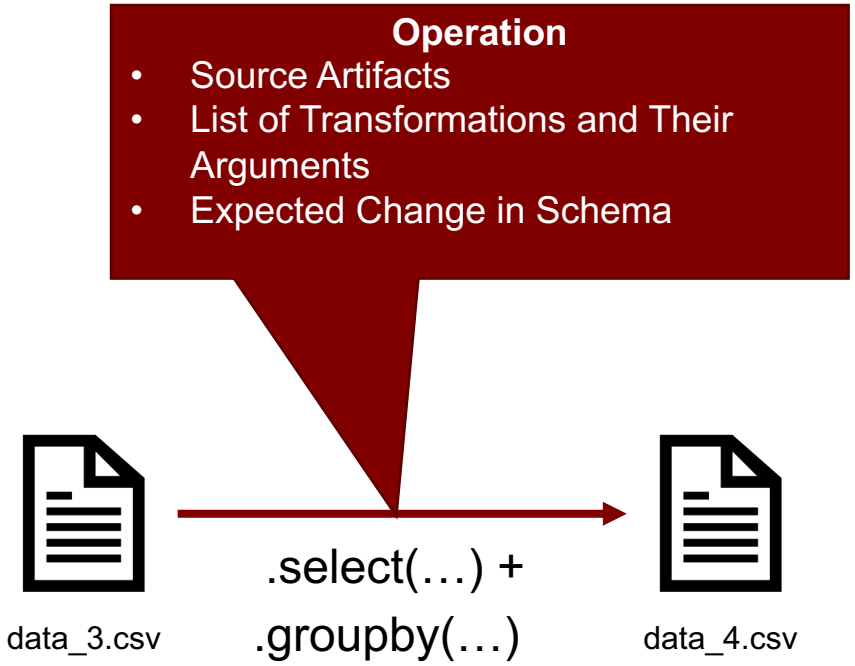
data_1.parquet

- Utilizes Python faker library
- Allow for generation for realistic datasets with varied datatypes
  - String, Numeric, DateTime etc.
  - Column Taxonomy: Groupable, Joinable, Numeric, String
- Future Work:
  - Generate data with FDs (Name->SSN)
  - Enforced Cardinalities and Distribution

{index}	iso8601	cryptocurrency_code	pyint	rn
0	2004-09-21T09:46:38	NEO	1453	10
1	2016-04-07T21:19:57	BCN	877	7
2	1973-08-09T20:35:50	USDT	8198	8
3	1985-08-24T17:07:41	EOS	7492	10
4	1979-06-16T21:01:12	NEM	157	6
5	2020-12-30T03:19:01	IOTA	5439	12
6	1995-05-03T04:56:00	BCH	2348	13
7	1972-09-02T20:03:53	XRP	1244	13
8	1990-12-27T10:33:05	ETC	8354	11
9	2017-07-03T20:19:32	WAVES	9717	11

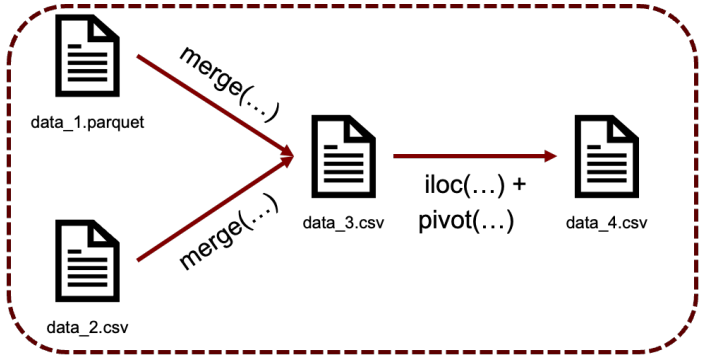
Operations

- Encodes operational semantics
- Constraints for the operation (minimum number of rows required, column types needed) etc.
- Associated arguments for the operation
- Expected Schema transformation
- Actual implementation / code generation left to client



Workflows

- A workflow consists of one or more “source artifacts” and a list of operations to generate new artifacts.
- JSON-Encoded Workflow Specifications
- Human-readable, editable
- Also generated by our workflow generator
- Future Work: Autogenerate JSON spec from existing Jupyter Notebooks




```
{
  "name": ”test_workflow",
  "operation_list": [{
    "sources": [
      “data_1.parquet”,
      “data_2.csv"
    ],
    "new_label": “data_3.csv",
    ”transformation_list": [{
      "op": ”merge",
      "args": {
        ”key": "VendorID”
      }
    }]},
  ...
  ]}
}
```

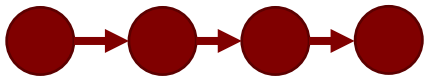
Workflow Generation Parameters

Parameter	Description
n	Number of Artifacts
r	Base Artifact Number of Rows
c	Base Artifact Number of Columns
b	Workflow Branching Factor
T	Set of Allowed Transformations
m	Materialization Rate

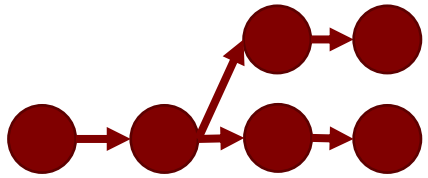
Table 3: Parameters for generating synthetic workflows.

- Allow for generation of varied types of workflows
- Branch Factor b:
  - Function that selects next artifact i to be manipulated with Probability

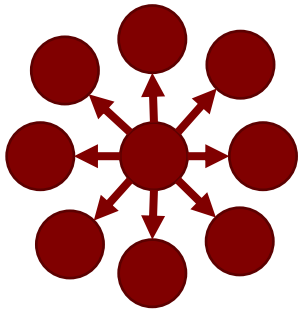
$$P[i] = \left(\frac{b}{e^{bn'} - 1}\right) e^{bi}$$



$b = 0.001$



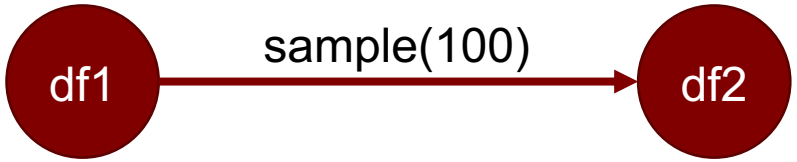
$b = 1.0$



$b = -1.0$

Current Client Implementations

- **Pandas**
- Modin
- SQLite

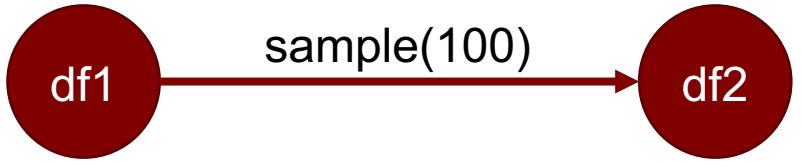


Generated Code

df2 = df1.sample(n=100)

Current Client Implementations

- Pandas
- Modin
- **SQLite**



Generated Code

```
CREATE VIEW df2 AS (SELECT *
FROM df1 ORDER BY RANDOM()
LIMIT 100)
```

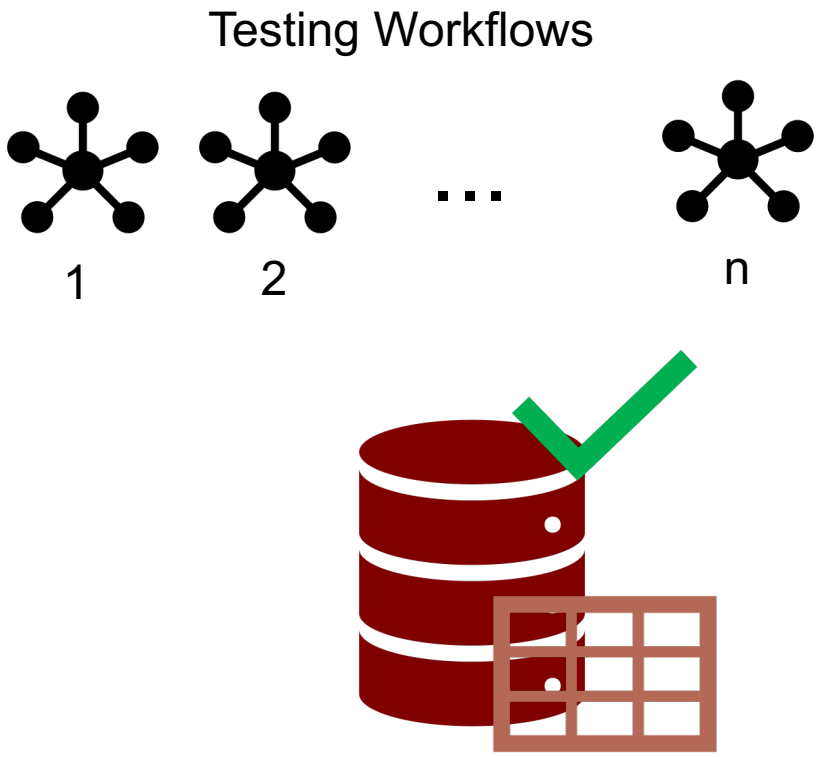
Current Client Implementations

Transformation	Description	Generation Constraints	pandas	SQLite	modin
load	Load (de-serialize) an artifact from a filesystem location	–	✓	✓	✓
select	Selects rows based on a filter condition	numeric ≥ 1	✓	✓	✓
apply	Create a new derived column as a scalar function applied to an existing numeric column	numeric > 1	✓	✓	✓
project	Project a set of columns	–	✓	✓	✓
sample	Randomly select rows from the artifact	–	✓	✓	✓
join	Inner Join with another artifact based on a key column	joinable ≥ 1	✓	✓	✓
pivot	Pivot the artifact by index, column and values	groupable ≥ 2; numeric ≥ 1	✓	–	✓
groupby	Groupby set of group_columns and apply an aggregate function on another set of agg_columns	groupable ≥ 1	✓	✓	✓
fill	Replace an old value in a column with another value	–	✓	–	✓
materialize	Execute stacked transformations to produce a new artifact	–	✓	✓	✓
serialize	Dump the contents of an artifact to disk	–	✓	✓	✓

Table 1: Transformation Implementation Matrix. The *Generation Constraints* column lists the minimum number of columns of each type required to generate each transformation using the random workflow generator.

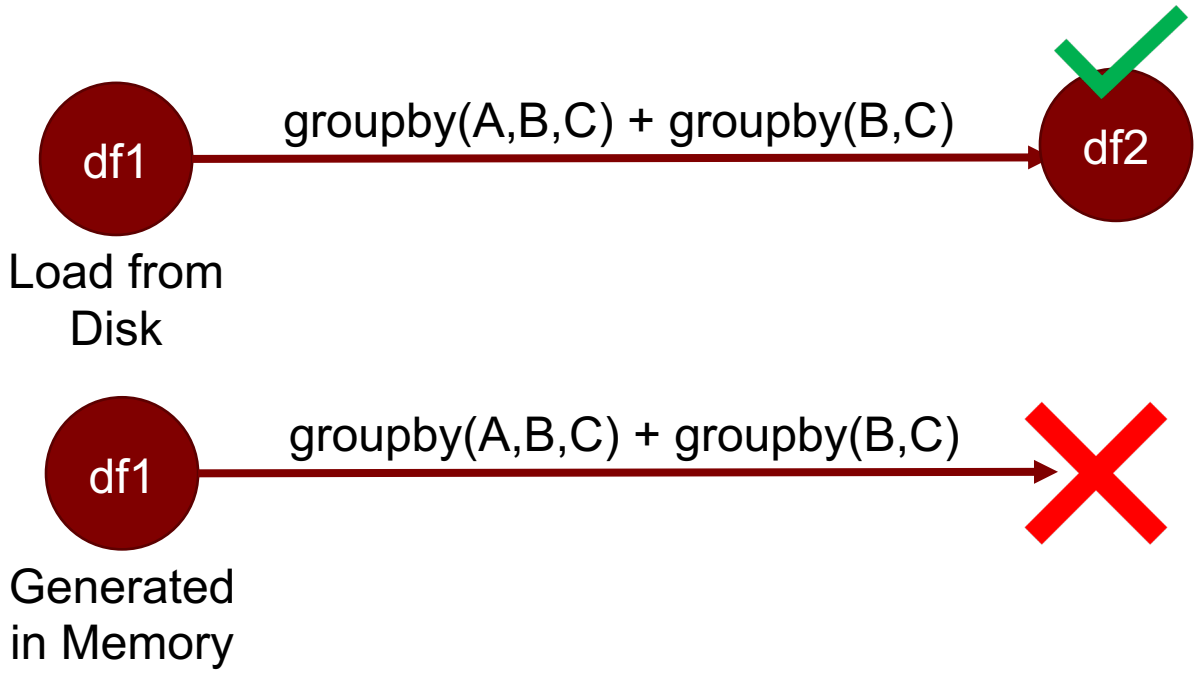
Fuzzy Data Use Cases

- **Performance and Correctness Analysis**
  - **Fuzzy Testing of DataFrame Systems**



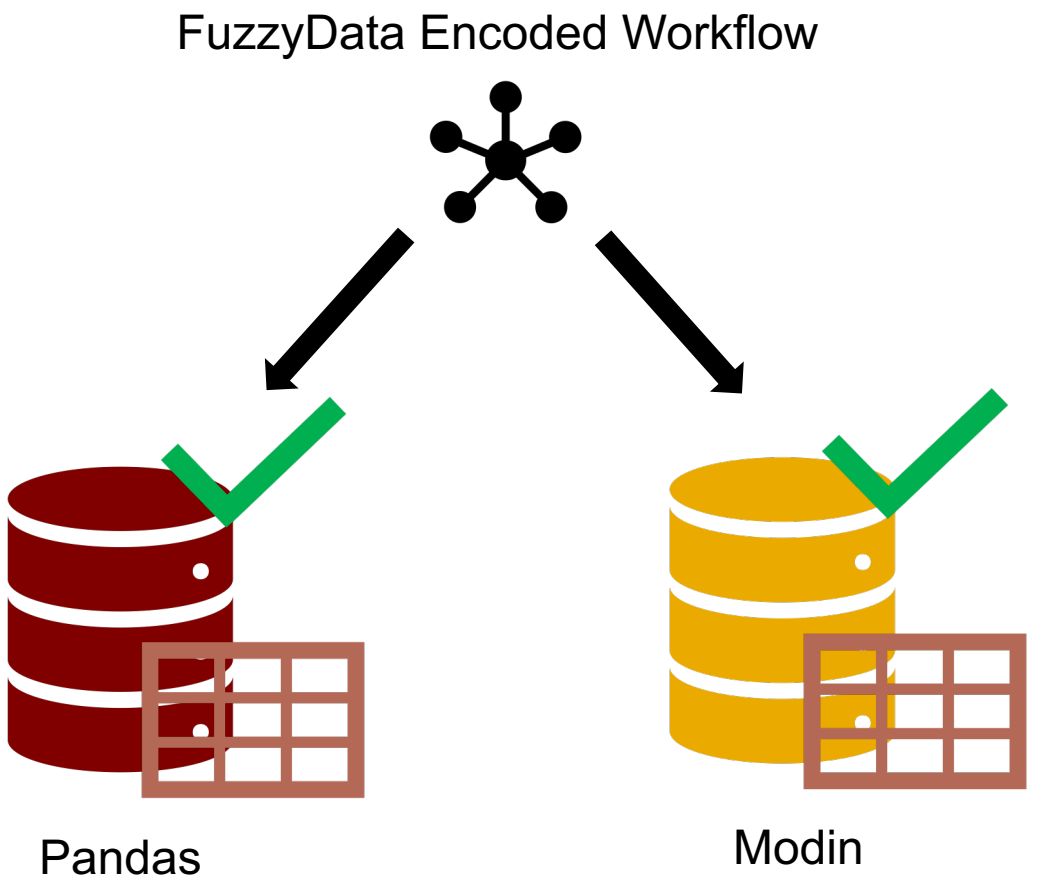
Fuzzy Testing of Data Frame Systems

- Test many workflows with varied datatypes and structures
- Use the automated test suite in FuzzyData to autogenerate test workflows
- Found a correctness issue in Modin, where multiple group-bys in memory fails to execute.

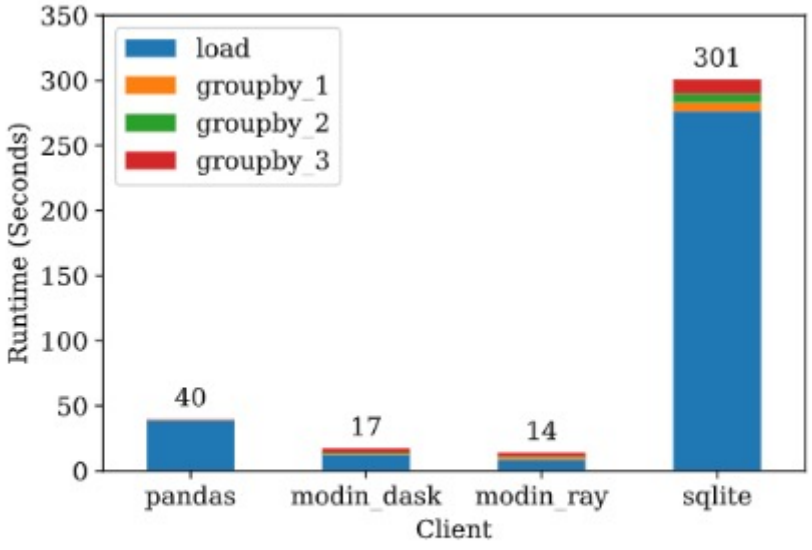


Fuzzy Data Use Cases

- Performance and Correctness Analysis
  - Fuzzy Testing of DataFrame Systems
- **Encoding and Replay of Workflows on Different Systems**



Encoding and Replay of Existing Workflows

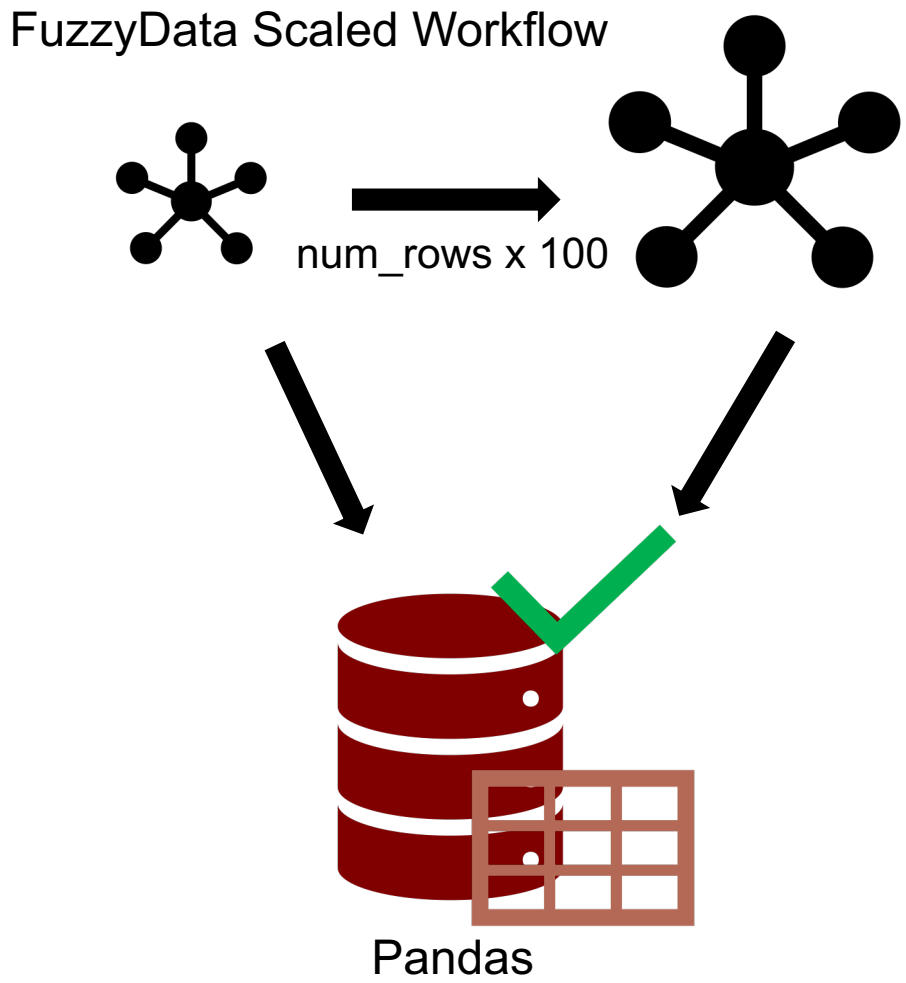


(a) Real-World Workflow Replay

- Encoded the NYC-Cab-based workflow (Modin's demo) and ran in 4 configurations
- Workflow shown by Modin to be performant using parallel loading and gropupby operations
- Performance gain matches claims made by Modin

Fuzzy Data Use Cases

- Performance and Correctness Analysis
  - Fuzzy Testing of DataFrame Systems
- Encoding and Replay of Workflows on Different Systems
- **Workload Scaling and Analysis**



Running a Randomized Workload at Scale

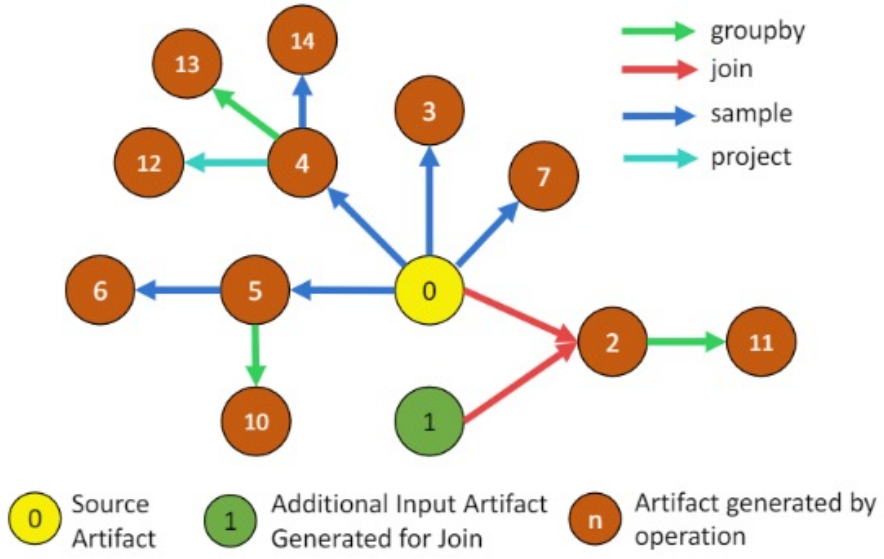
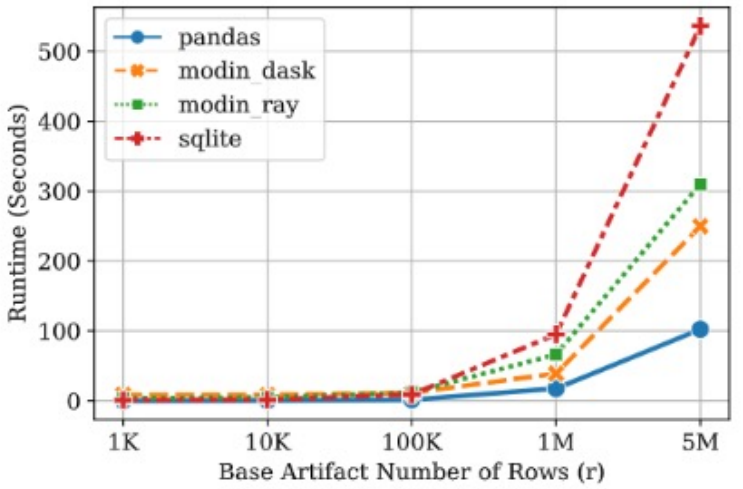
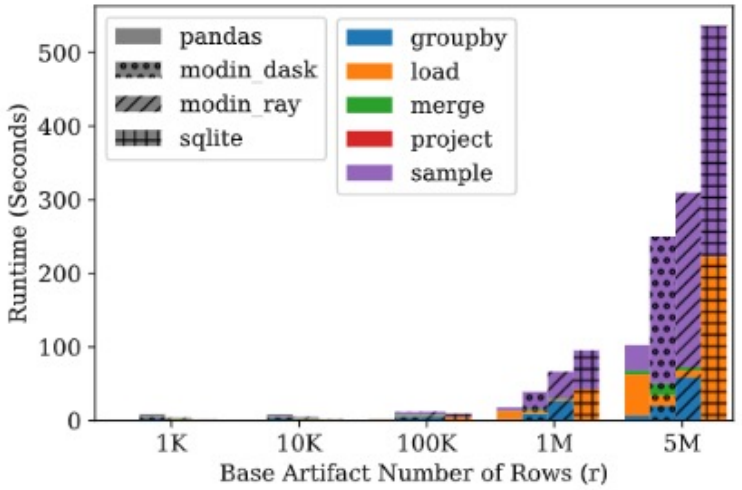


Figure 2: DAG of a randomly generated workflow using the generator described in Section 3.4. The generation parameters used were $(n = 15, r = 1000, c = 20, b = 0.01, m = 1)$.



(b) Scaling Experiment -Total Runtime



(c) Scaling Experiment - Breakdown of runtime.

Easy to Use, CLI Interface and Open Source

```
# Install via pip
$ pip install fuzzydata
# Generate a Random Pandas Workflow with 20 artifacts
$ fuzzydata --wf_client=pandas --output_dir=/tmp/fd_test/
--columns=20 --rows=10000 --versions=20
# Replay the Workflow in Modin
$ fuzzydata --wf_client=modin --replay_dir=/tmp/fd_test/
--output_dir=/tmp/fd_test_modin/
```

https://github.com/suhailrehman/fuzzydata

Future Work

- Use machine learning models to generate realistic sequences of operations [AutoSuggest'18]
- Generation for Realistic FDs between columns (Say, Name/SSN) etc.
- Additional Clients to be implemented.
  - Polars
  - RAPIDS
  - etc..

Takeaways:

1. Relational Benchmarks not a good fit for dataframe systems
2. Fuzzydata can help in
    1. generating/specifying workloads
    2. replaying and scaling workloads among different dataframe clients
    3. Enable fuzzy testing of systems to catch performance and correctness bugs

Mohammed Suhail Rehman

suhail@uchicago.edu

www.suhailrehman.com

https://github.com/suhailrehman/fuzzydata