

Pandas on DuckDB With Ponder

Aditya Parameswaran
Cofounder and President, Ponder
Associate Professor, UC Berkeley



 **pandas** The Swiss Army Knife of Data Science/ML



The Swiss Army Knife of Data Science/ML

A screenshot of a web browser displaying an article from The Register. The browser's address bar shows the URL 'theregister.com/2017/09/14/python_explosion_blamed_on_pandas/'. The page has a red header with 'The Register' logo and a search icon. Below the header, there is a red bar with 'SIGN IN' and a search icon. The main content area has a red bar with '{* DEVOPS *}' and the article title 'Python explosion blamed on pandas'. The subtitle is 'Data science fad just won't die'. The author is 'Thomas Claburn in San Francisco' and the date is 'Thu 14 Sep 2017 // 20:02 UTC'. The article text starts with 'Not content to bait developers by declaring that Python is the fastest-growing major programming language, coding community site Stack Overflow has revealed the reason for its metastasis.' There are 34 comments and a share icon.





The Swiss Army Knife of Data Science/ML

the register.com/2017/09/14/python_explosion_blamed_on_pandas/

SIGN IN The Register

{* DEVOPS *}

Python explosion blamed on pandas

Data science fad just won't die

Thomas Claburn in San Francisco Thu 14 Sep 2017 // 20:02 UTC

34

Not content to bait developers by declaring that Python is the **fastest-growing major programming language**, coding community site Stack Overflow has revealed the reason for its metastasis.

Pandas Is Now As Popular As Python Was in 2016

Peter Olson
Jul 29, 2022 • 10 min read

ARTICLES



pandas The Swiss Army Knife of Data Science/ML



the register.com/2017/09/14/python_explosion_blamed_on_pandas/

SIGN IN The Register

{* DEVOPS *}

Python explosion blamed on pandas

Data science fad just won't die

Thomas Claburn in San Francisco Thu 14 Sep 2017 // 20:02 UTC

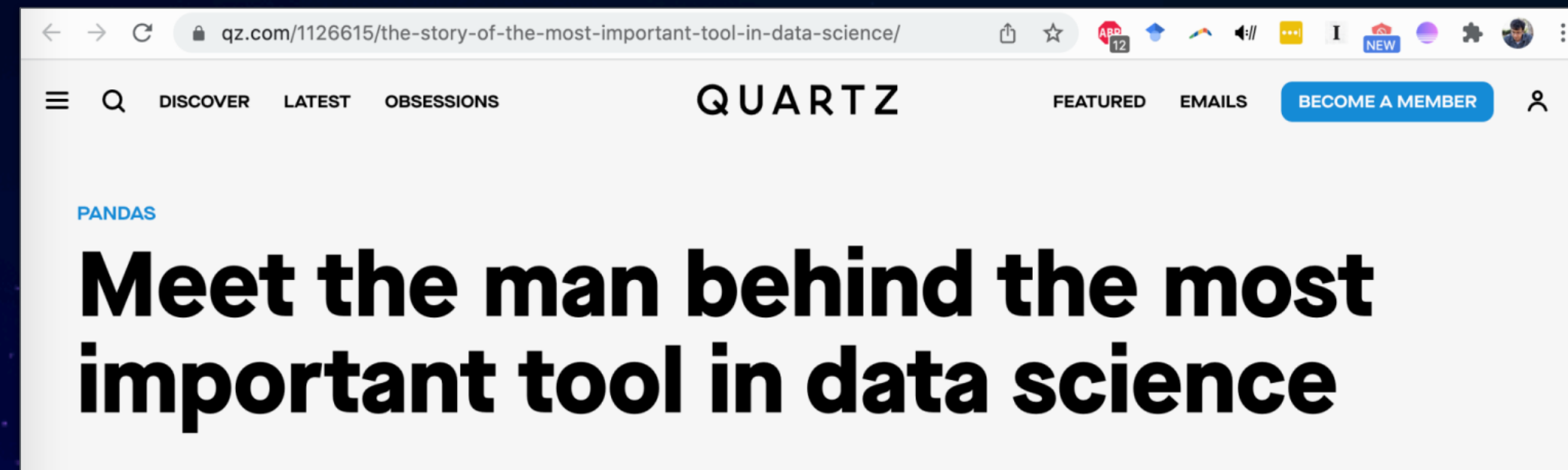
34

Not content to bait developers by declaring that Python is the **fastest-growing major programming language**, coding community site Stack Overflow has revealed the reason for its metastasis.

Pandas Is Now As Popular As Python Was in 2016

Peter Olson
Jul 29, 2022 • 10 min read

ARTICLES

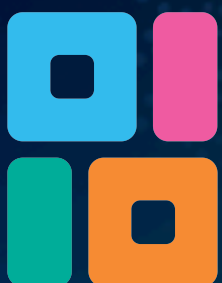


qz.com/1126615/the-story-of-the-most-important-tool-in-data-science/

DISCOVER LATEST OBSESSIONS QUARTZ FEATURED EMAILS BECOME A MEMBER

PANDAS

Meet the man behind the most important tool in data science



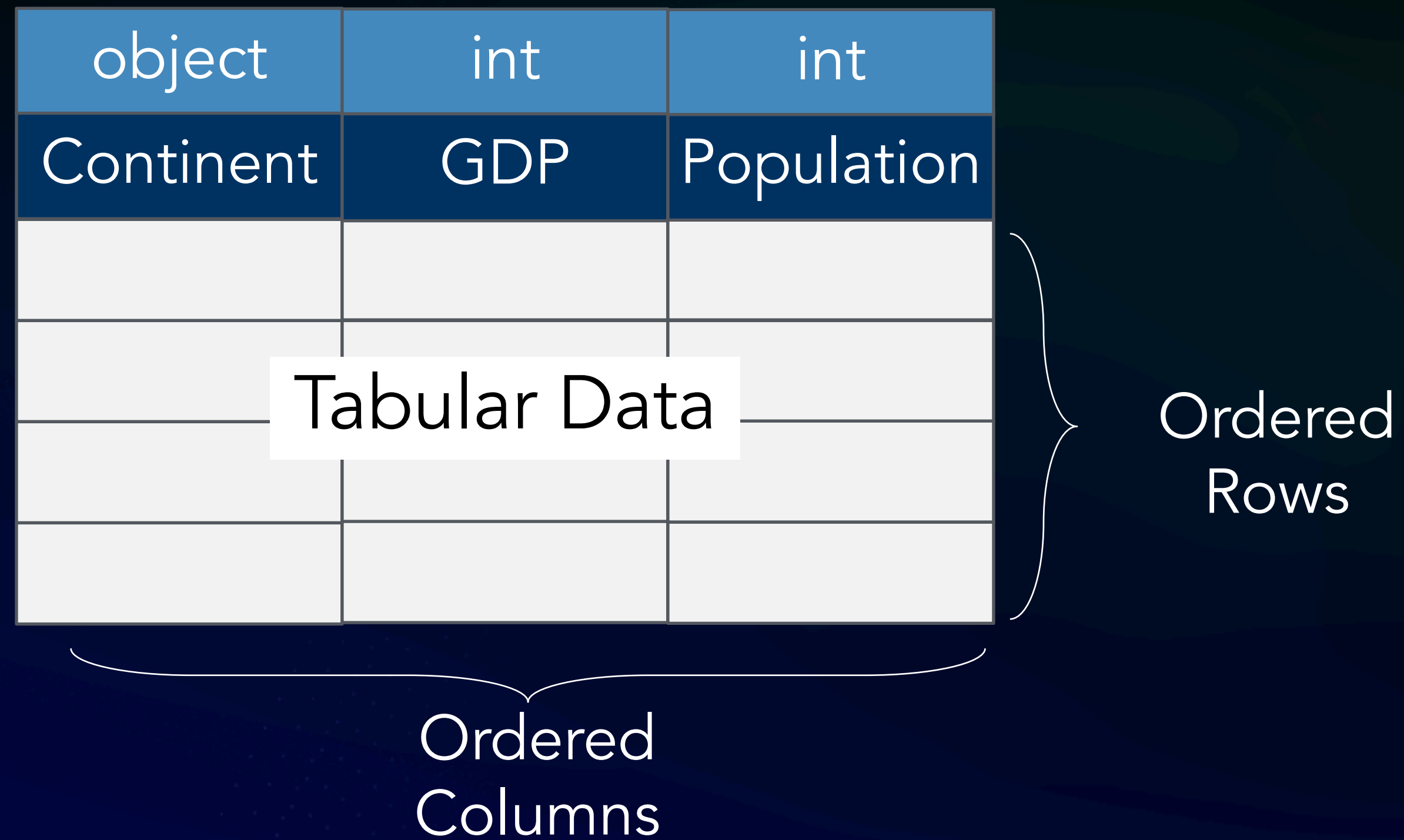
Why? A Flexible & Convenient Dataframe Data Model

object	int	int
Continent	GDP	Population

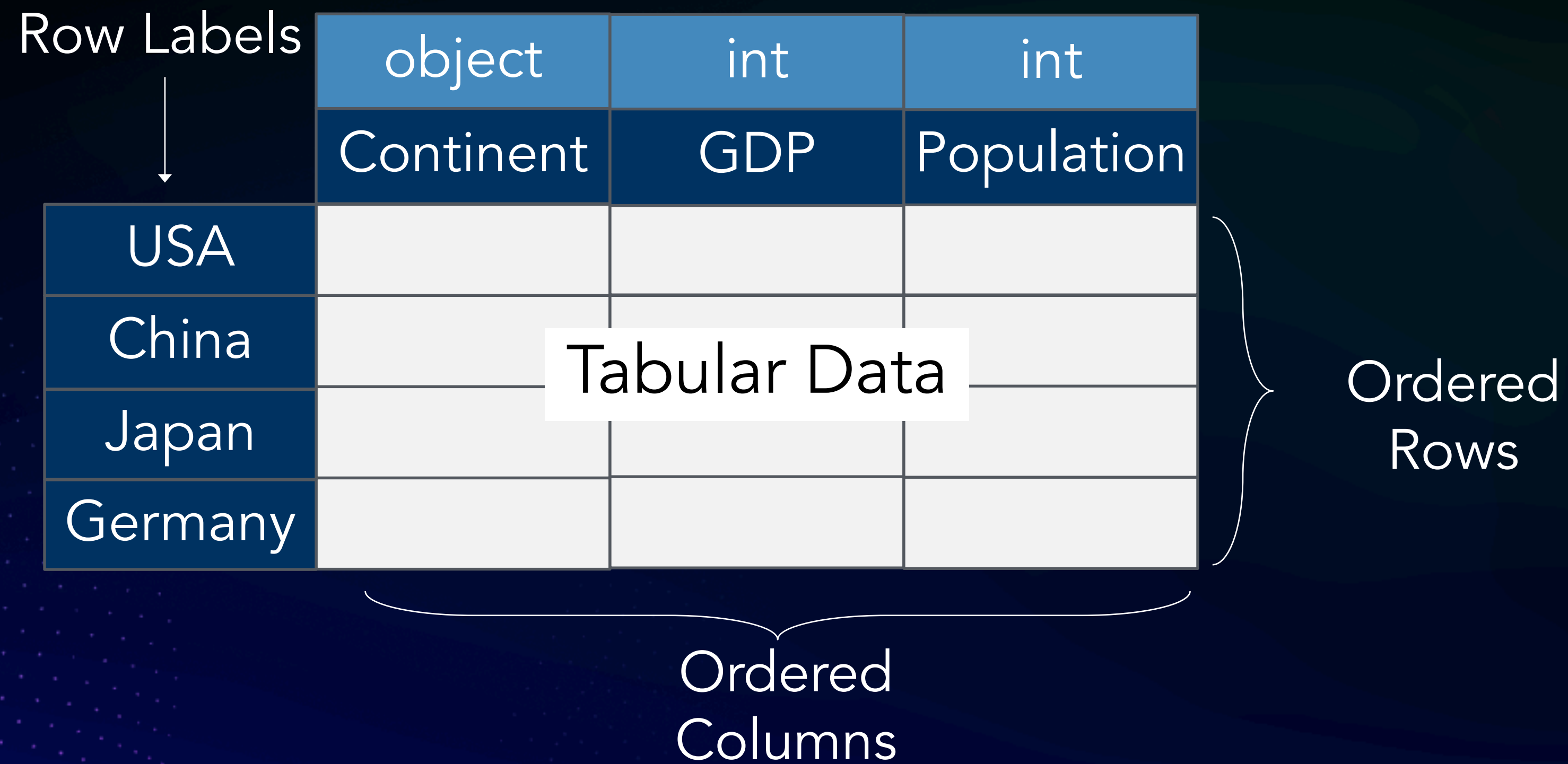
Tabular Data



Why? A Flexible & Convenient Dataframe Data Model



Why? A Flexible & Convenient Dataframe Data Model



Why? A Flexible & Convenient Dataframe Data Model

Schema not required upfront

Supports mixed types / column

Row Labels

	object	int	int
	Continent	GDP	Population
USA			
China			
Japan			
Germany			

Tabular Data

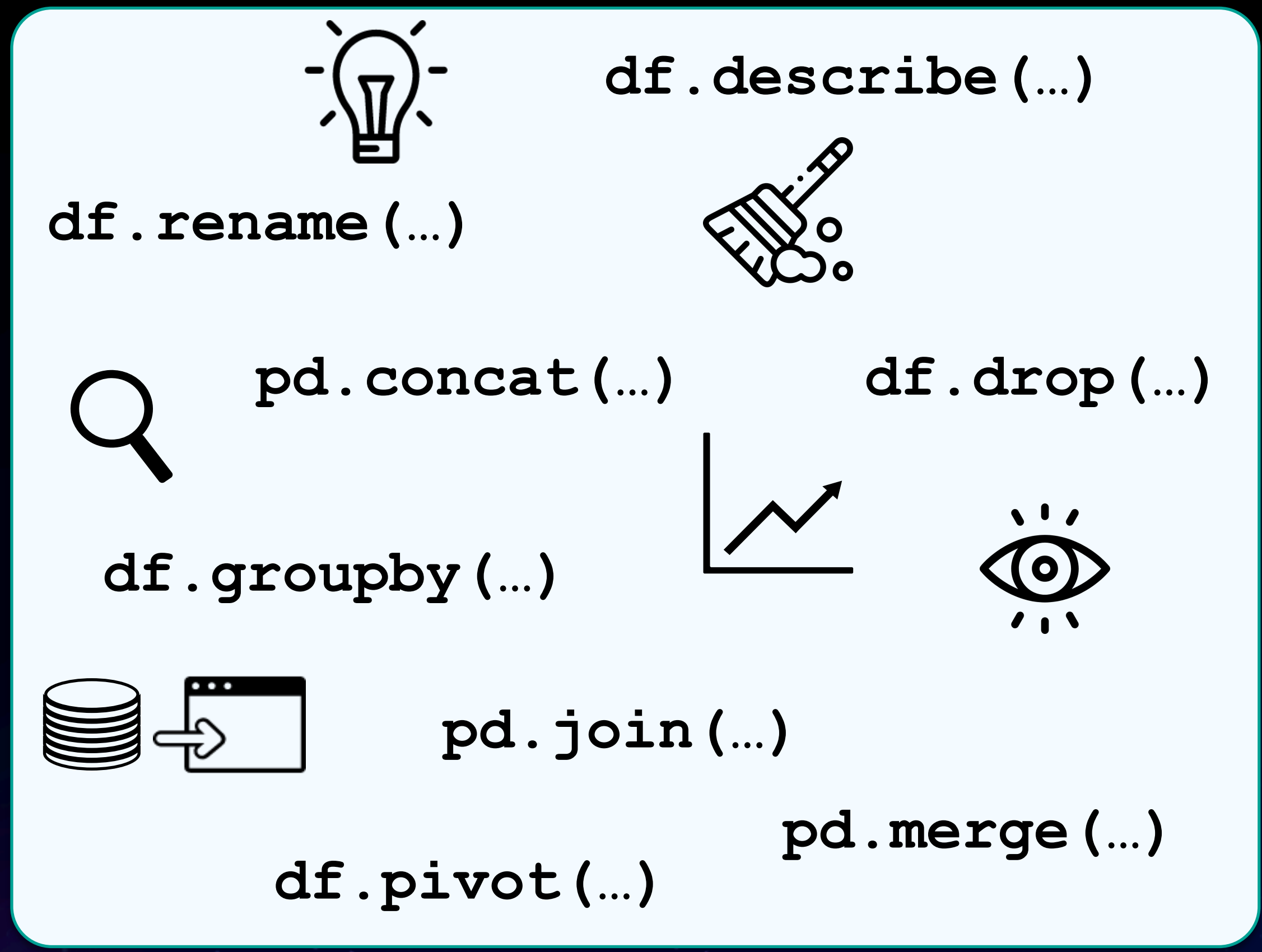
Ordered Rows

Ordered Columns


+ Hierarchical row/column labels (not shown)






Why? (II) An Expressive & Concise Data Science API


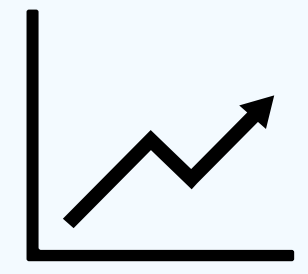



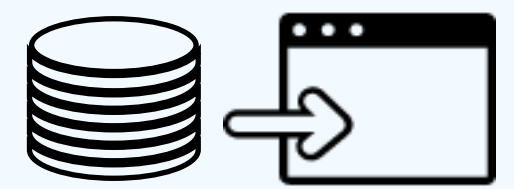
The image displays a collection of data science API functions and their corresponding icons, arranged in a grid-like fashion. The functions are: `df.describe(...)` (lightbulb icon), `df.rename(...)` (lightbulb icon), `pd.concat(...)` (magnifying glass icon), `df.drop(...)` (microscope icon), `df.groupby(...)` (magnifying glass icon), `df.pivot(...)` (line graph icon), `pd.join(...)` (database icon), and `pd.merge(...)` (eye icon).


 `df.describe(...)`

`df.rename(...)` 

 `pd.concat(...)`  `df.drop(...)`


`df.groupby(...)`    `df.drop(...)`


 `pd.join(...)`


`df.pivot(...)` `pd.merge(...)` 

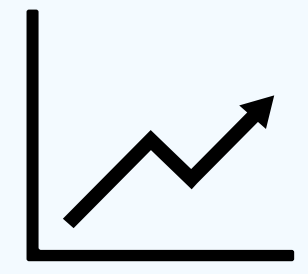

600+ functions to clean, featurize, explore, and summarize data spanning rel., linear, & spreadsheet algebra

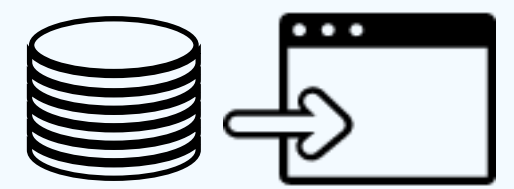
Why? (II) An Expressive & Concise Data Science API

 `df.describe (...)`

`df.rename (...)` 

 `pd.concat (...)` `df.drop (...)`

`df.groupby (...)`  

 `pd.join (...)`

`df.pivot (...)` `pd.merge (...)`

With operations such as:

- Drop columns with null values
- Transpose
- One hot encoding
- Positional updates
- Elementwise matrix ops
- ...

600+ functions to clean, featurize, explore, and summarize data spanning rel., linear, & spreadsheet algebra

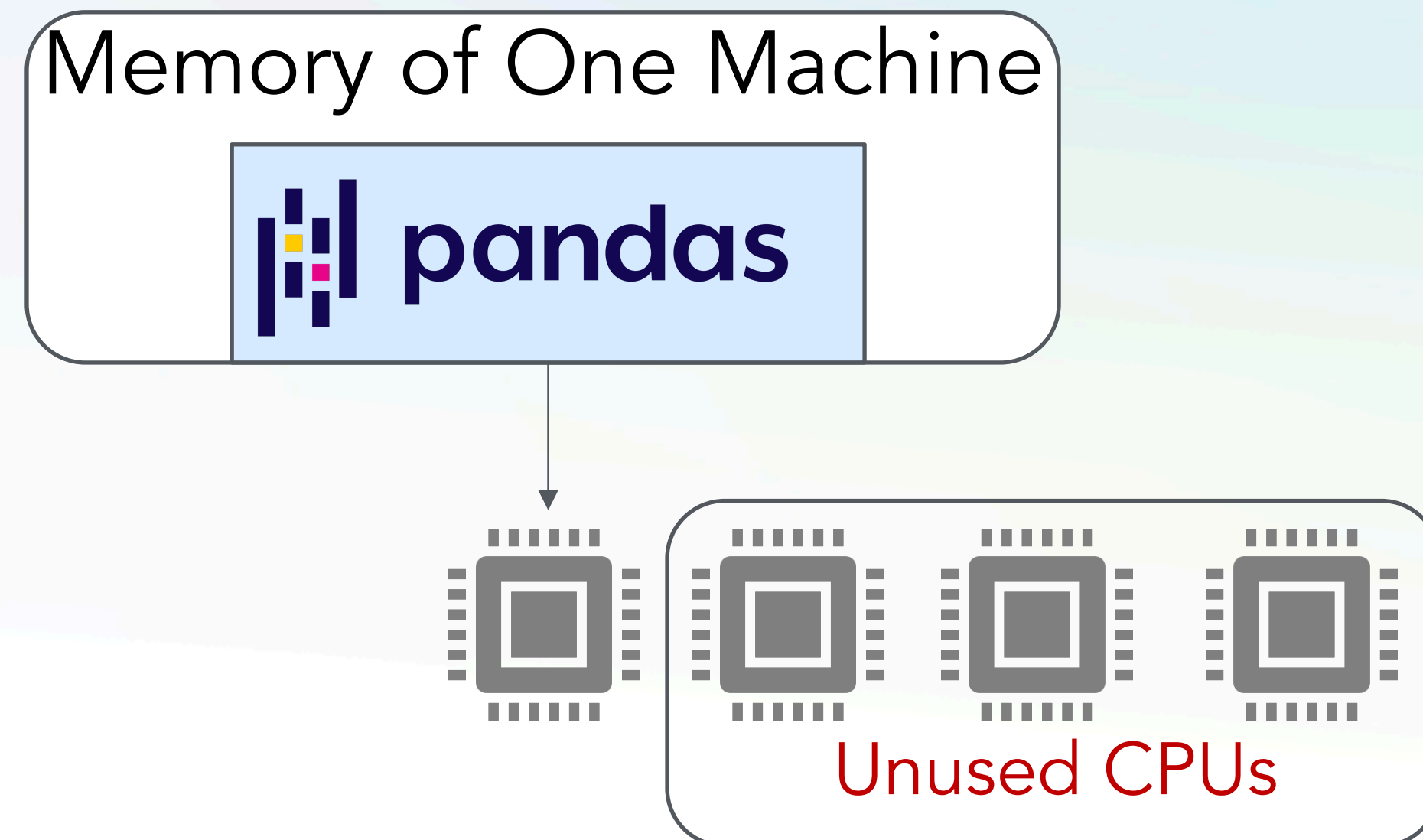
But Pandas Doesn't Scale!

🌀 Single threaded

But Pandas Doesn't Scale!

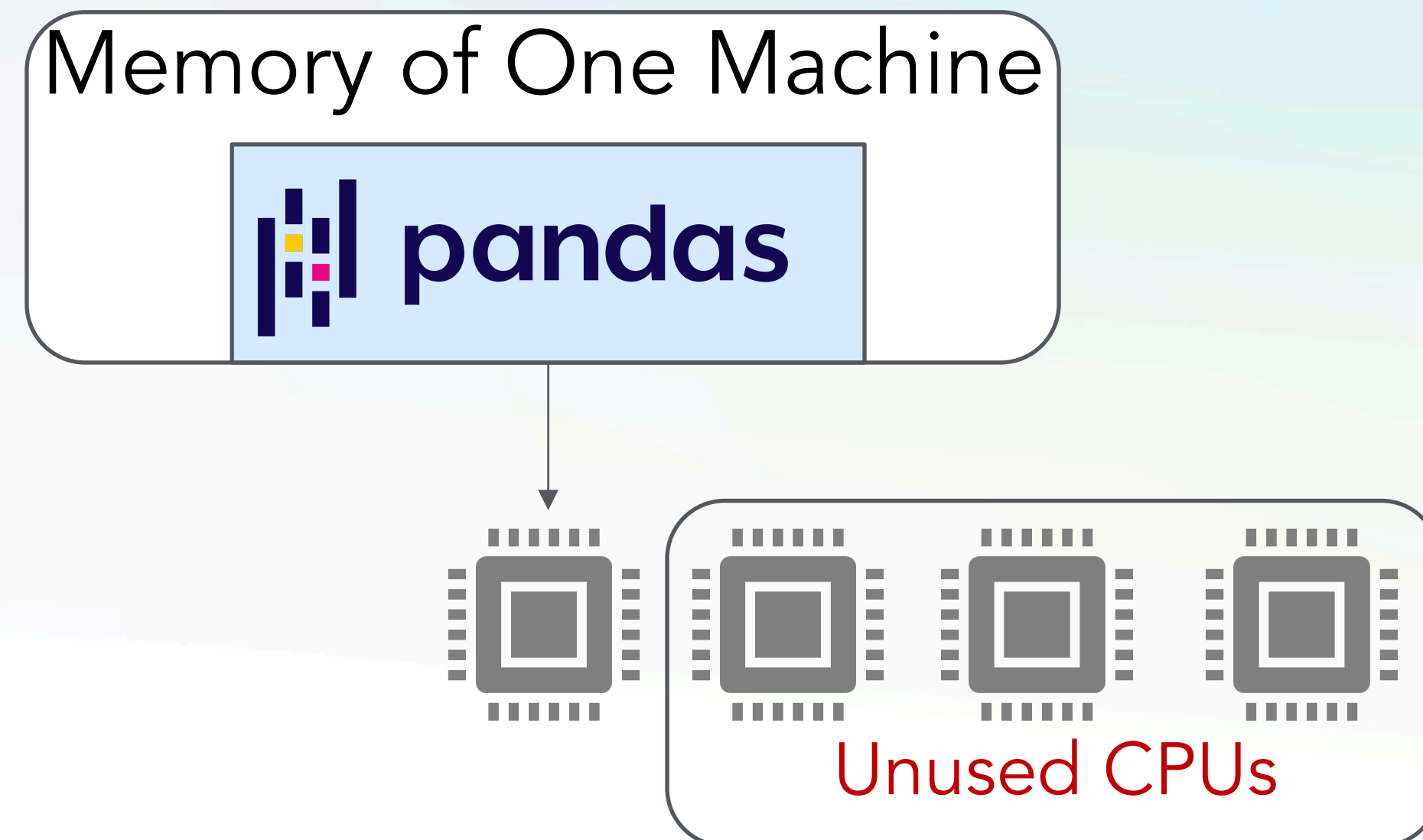
 Single threaded

 Dataset size restricted to memory of a machine



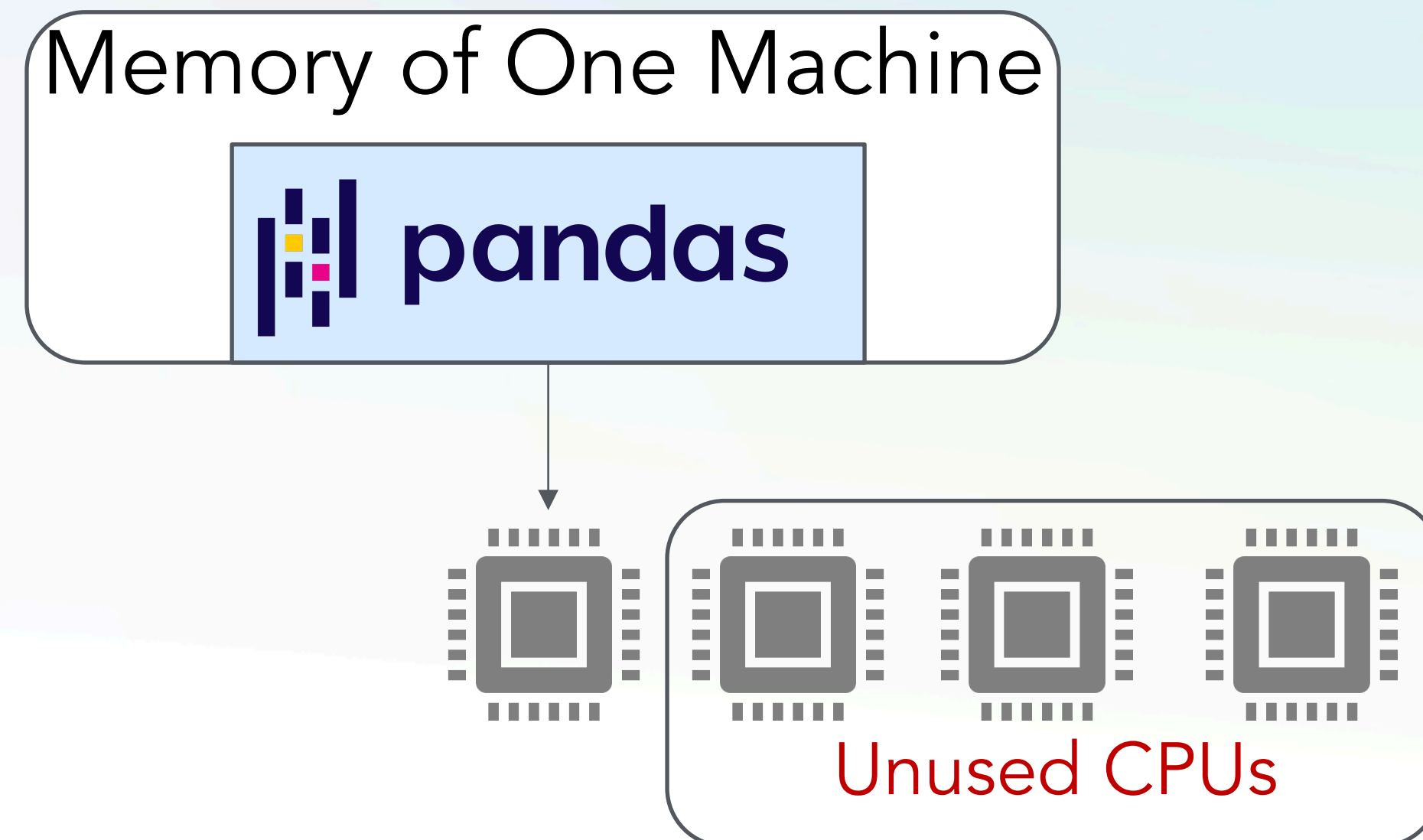
But Pandas Doesn't Scale!

- 🌀 Single threaded
- 📂 Dataset size restricted to memory of a machine
- 👛 Inefficient with memory



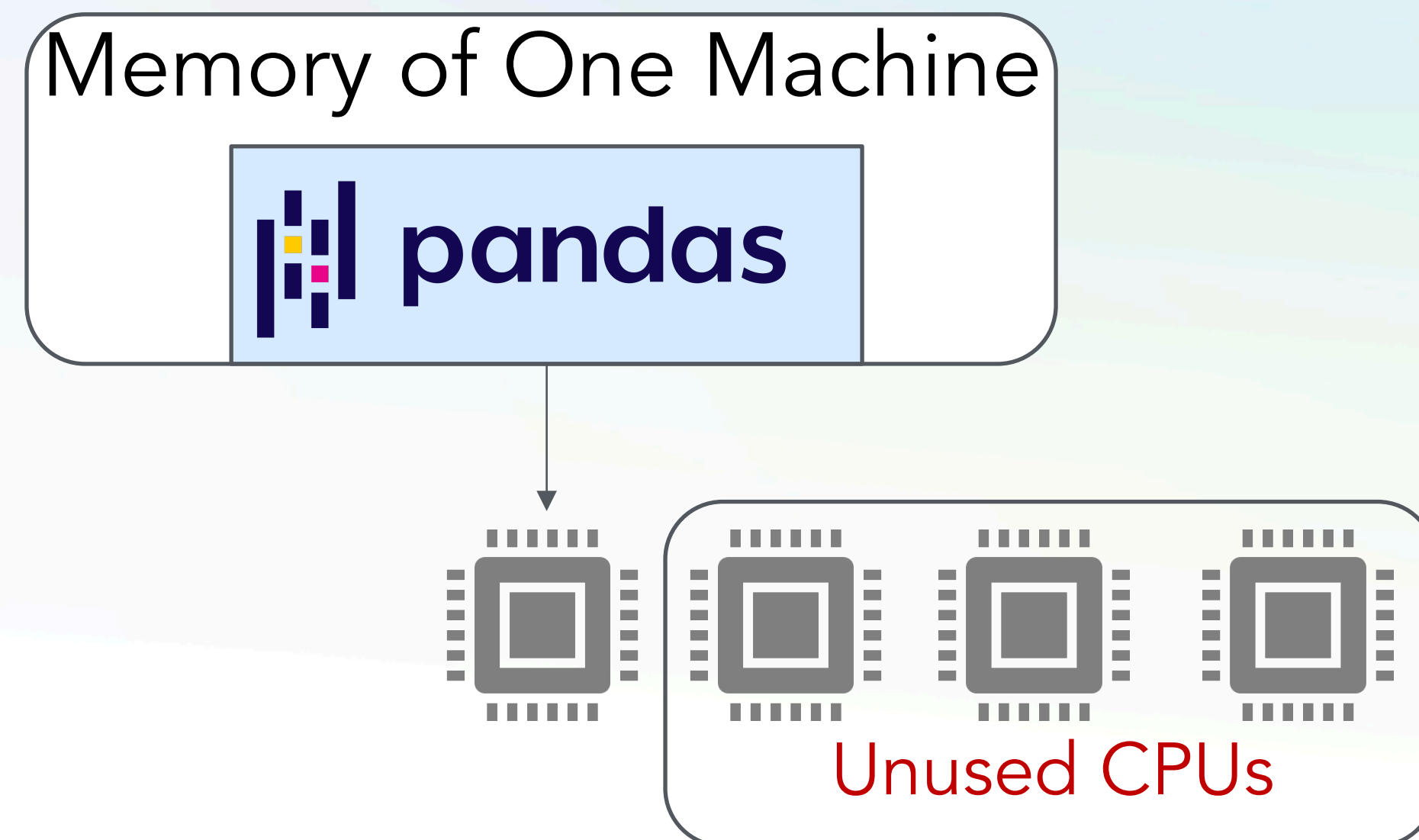
But Pandas Doesn't Scale!

- 🌀 Single threaded
- 📂 Dataset size restricted to memory of a machine
- 👛 Inefficient with memory
- 🧩 No real optimization



But Pandas Doesn't Scale!

- 🌀 Single threaded
- 📂 RAM Dataset size restricted to memory of a machine
- 👛 Inefficient with memory
- 🧩 No real optimization



MODIN A "Drop-in" Scalable Version of Pandas

```
# import pandas as pd  
import modin.pandas as pd
```

MODIN A "Drop-in" Scalable Version of Pandas

```
# import pandas as pd  
import modin.pandas as pd
```

pandas API

 MODIN



Ray



Dask

...

<https://github.com/modin-project/modin>

MODIN A "Drop-in" Scalable Version of Pandas

```
# import pandas as pd
import modin.pandas as pd
```

pandas API

 MODIN



Ray



Dask

...

- 🔥 10+ Million Downloads (1M+/month)
- 🤝 100+ contributors
- 🚀 Used by 40+ companies & orgs
- 🎯 Part of AWS, Intel distros

<https://github.com/modin-project/modin>

MODIN A "Drop-in" Scalable Version of Pandas

```
# import pandas as pd
import modin.pandas as pd
```

pandas API

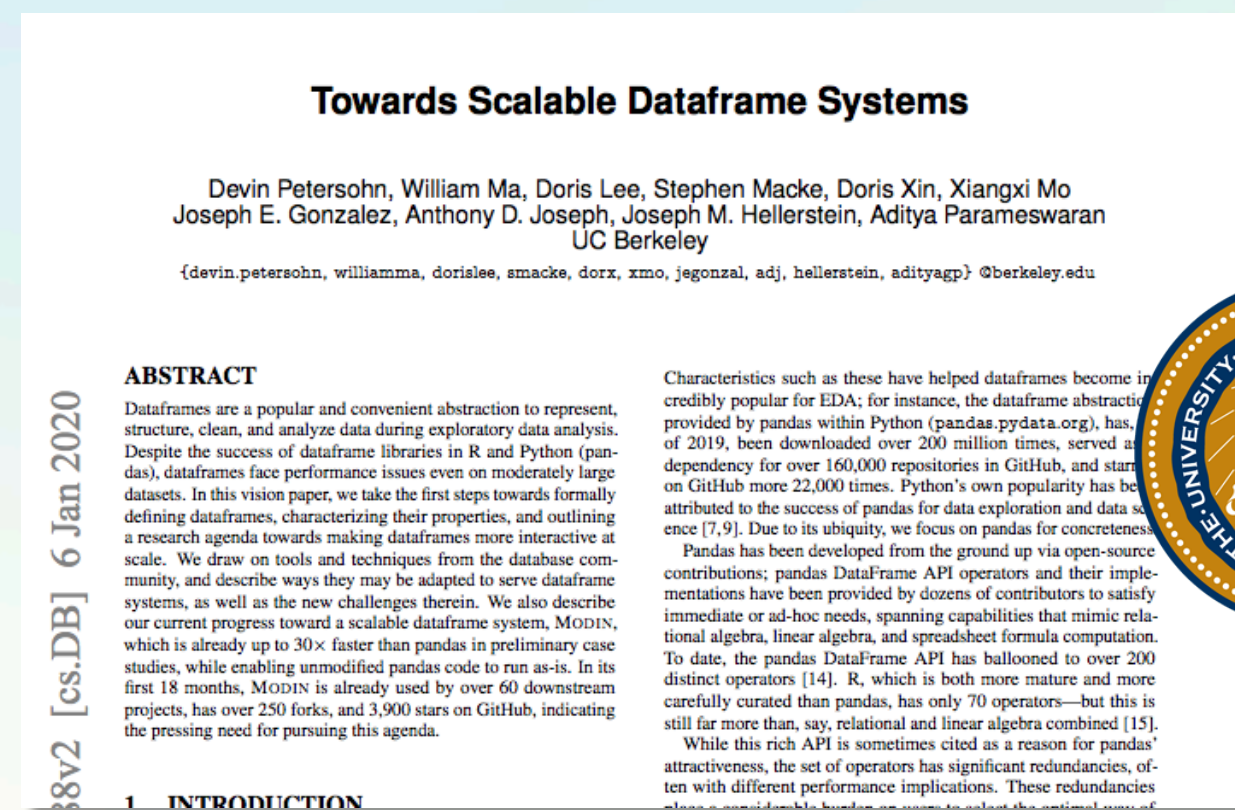
 MODIN



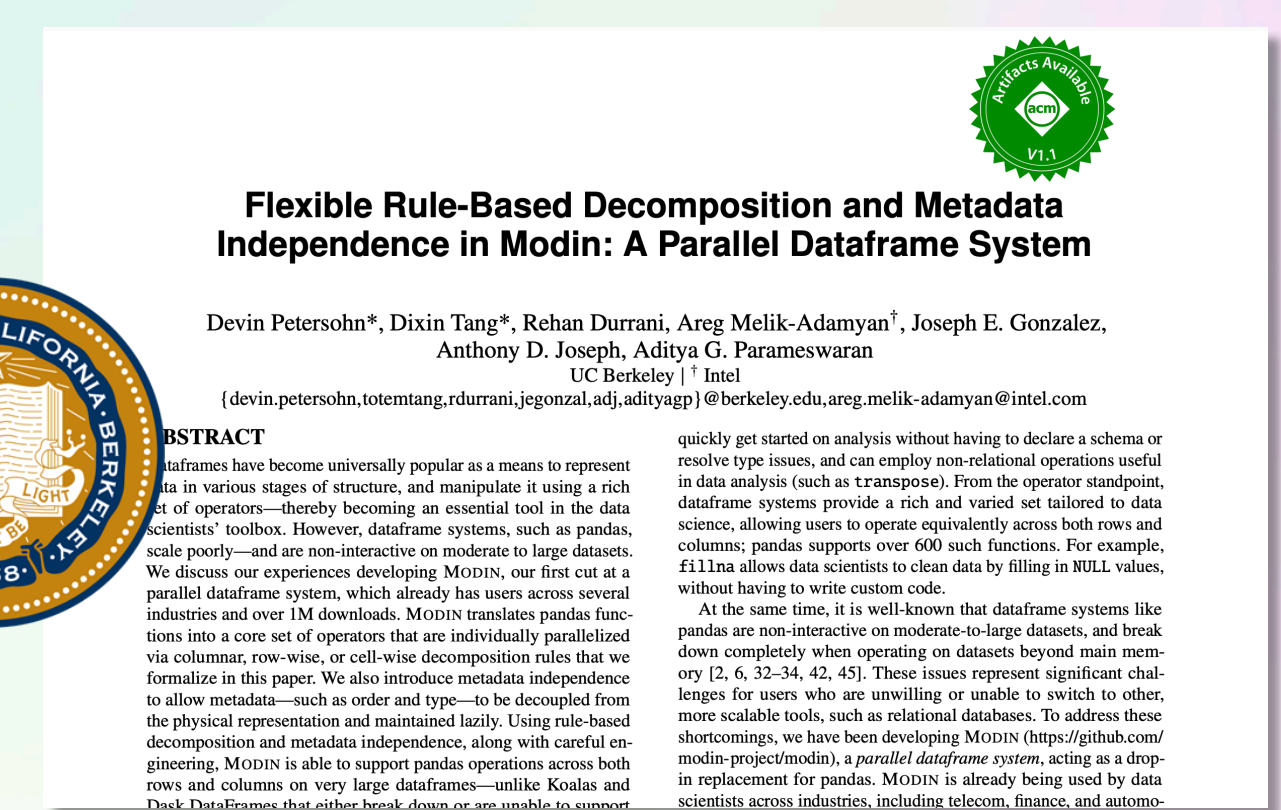
...

<https://github.com/modin-project/modin>

-  10+ Million Downloads (1M+/month)
-  100+ contributors
-  Used by 40+ companies & orgs
-  Part of AWS, Intel distros



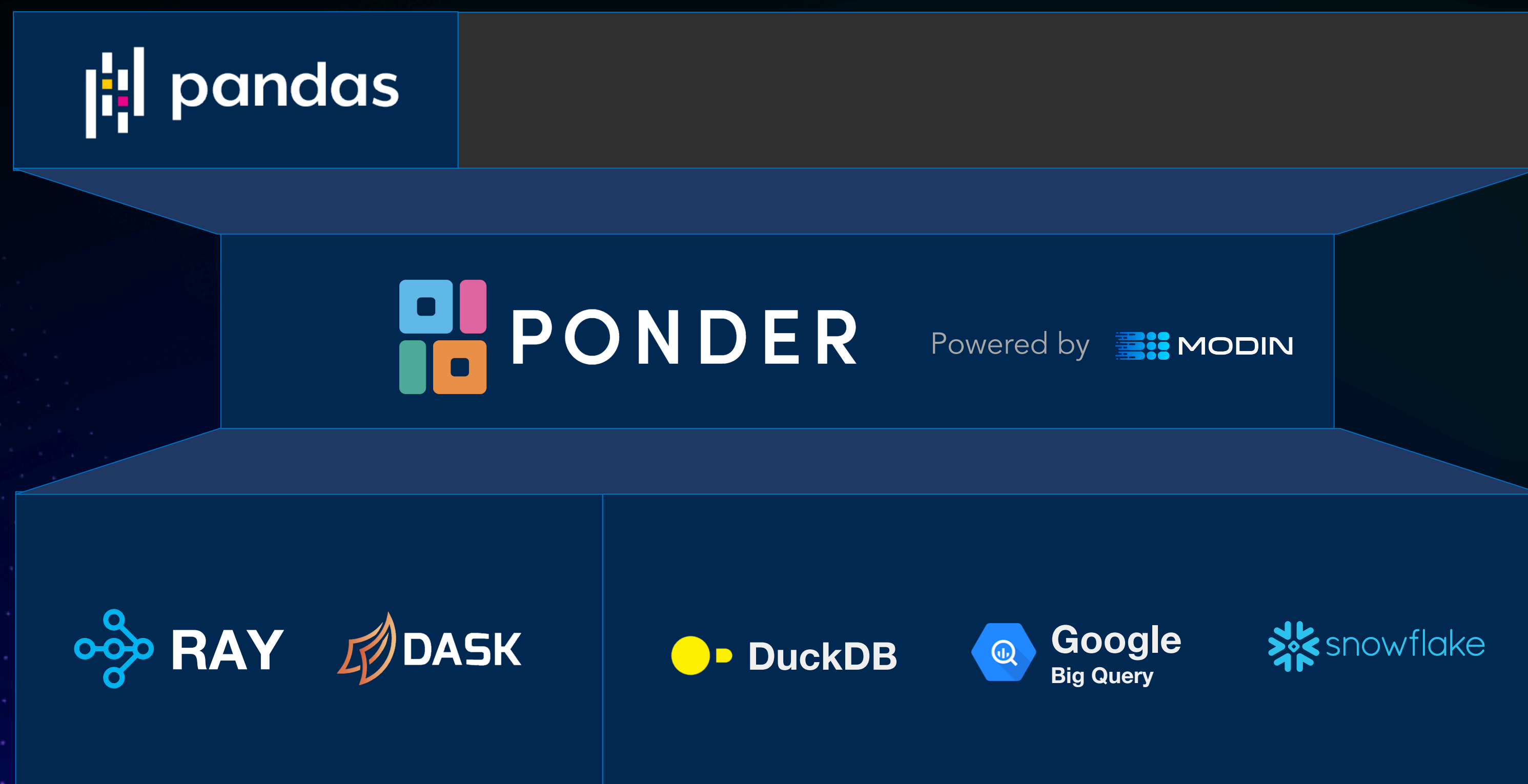
"Core" Dataframe Algebra



Parallelization Framework

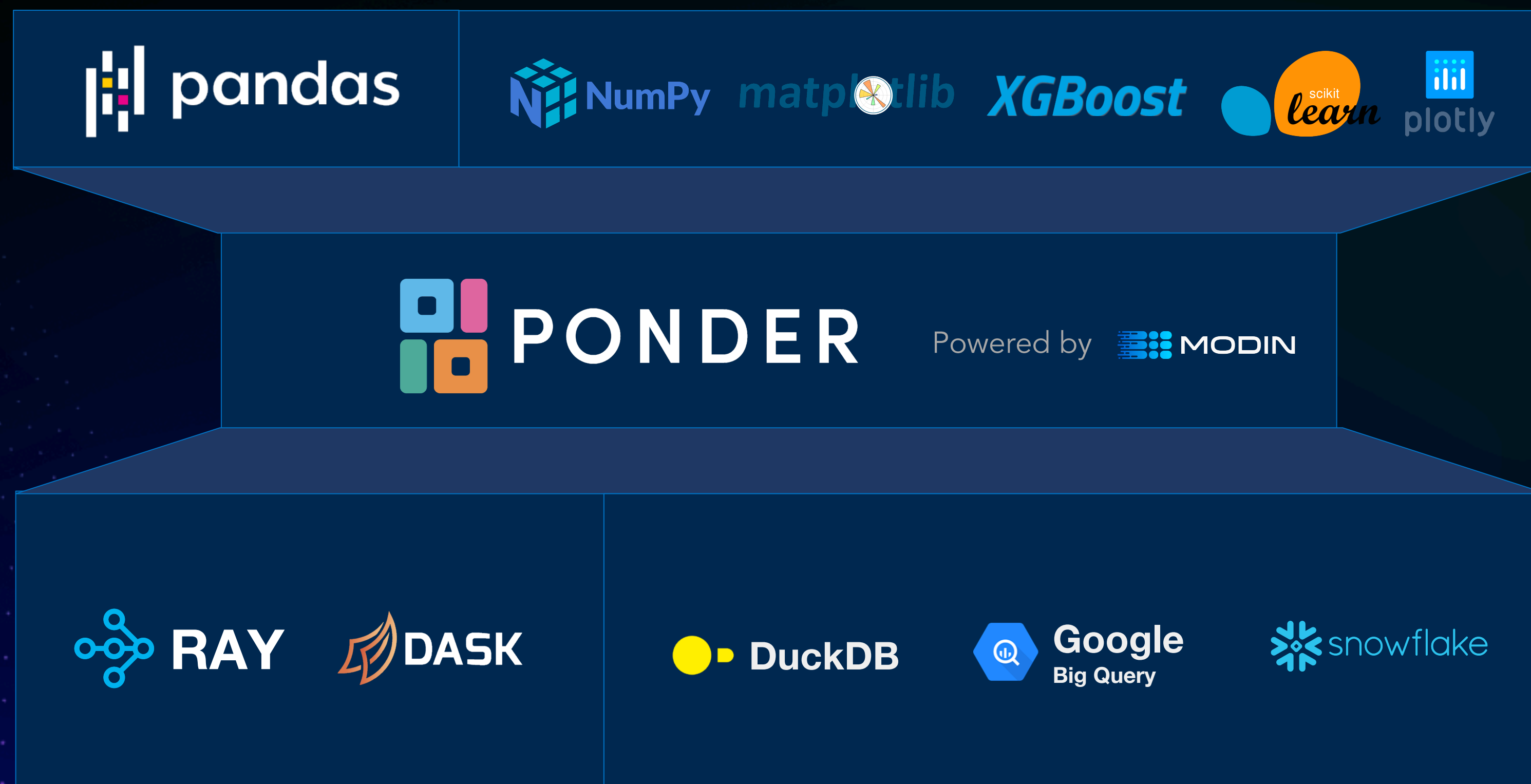


The Next Generation with Ponder : Data Science *at scale now powered by your database*





The Next Generation with Ponder : Data Science *at scale now powered by your database*





The Magic of Ponder

Min-Max normalization

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

One-line in pandas



```
(df - df.min()) / (df.max() - df.min())
```



The Magic of Ponder

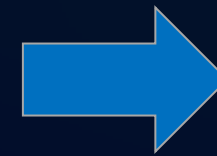
Min-Max normalization

$$x_{scaled} = \frac{x - \min(x)}{\max(x) - \min(x)}$$

One-line in pandas



```
(df - df.min()) / (df.max() - df.min())
```



Equivalent SQL is 300+ lines!!

```
1 SELECT |
2   _PONDER_ROW_LABELS_,
3   C_CUSTKEY,
4   C_NATIONKEY,
5   C_ACCTBAL
6 FROM
7   (
8     SELECT
9     *
10    FROM
11    (
12      SELECT
13      _PONDER_ROW_NUMBER_,
14      _PONDER_ROW_LABELS_,
15      C_CUSTKEY / C_CUSTKEY_ponder_right AS C_CUSTKEY,
16      C_NATIONKEY / C_NATIONKEY_ponder_right AS C_NATIONKEY,
17      C_ACCTBAL / C_ACCTBAL_ponder_right AS C_ACCTBAL
18     FROM
19     (
20       SELECT
21       _PONDER_ROW_NUMBER_,
22       _PONDER_ROW_LABELS_,
23       C_CUSTKEY - C_CUSTKEY_ponder_right AS C_CUSTKEY,
24       C_NATIONKEY - C_NATIONKEY_ponder_right AS C_NATIONKEY,
25       C_ACCTBAL - C_ACCTBAL_ponder_right AS C_ACCTBAL
26     FROM
27     (
28       SELECT
29       _PONDER_ROW_NUMBER_,
30       _PONDER_ROW_LABELS_,
31       C_CUSTKEY,
32       C_NATIONKEY,
33       C_ACCTBAL
34     FROM
```


Easy to Get Started

Create a database connection

```
import duckdb
db_con = duckdb.connect()
```

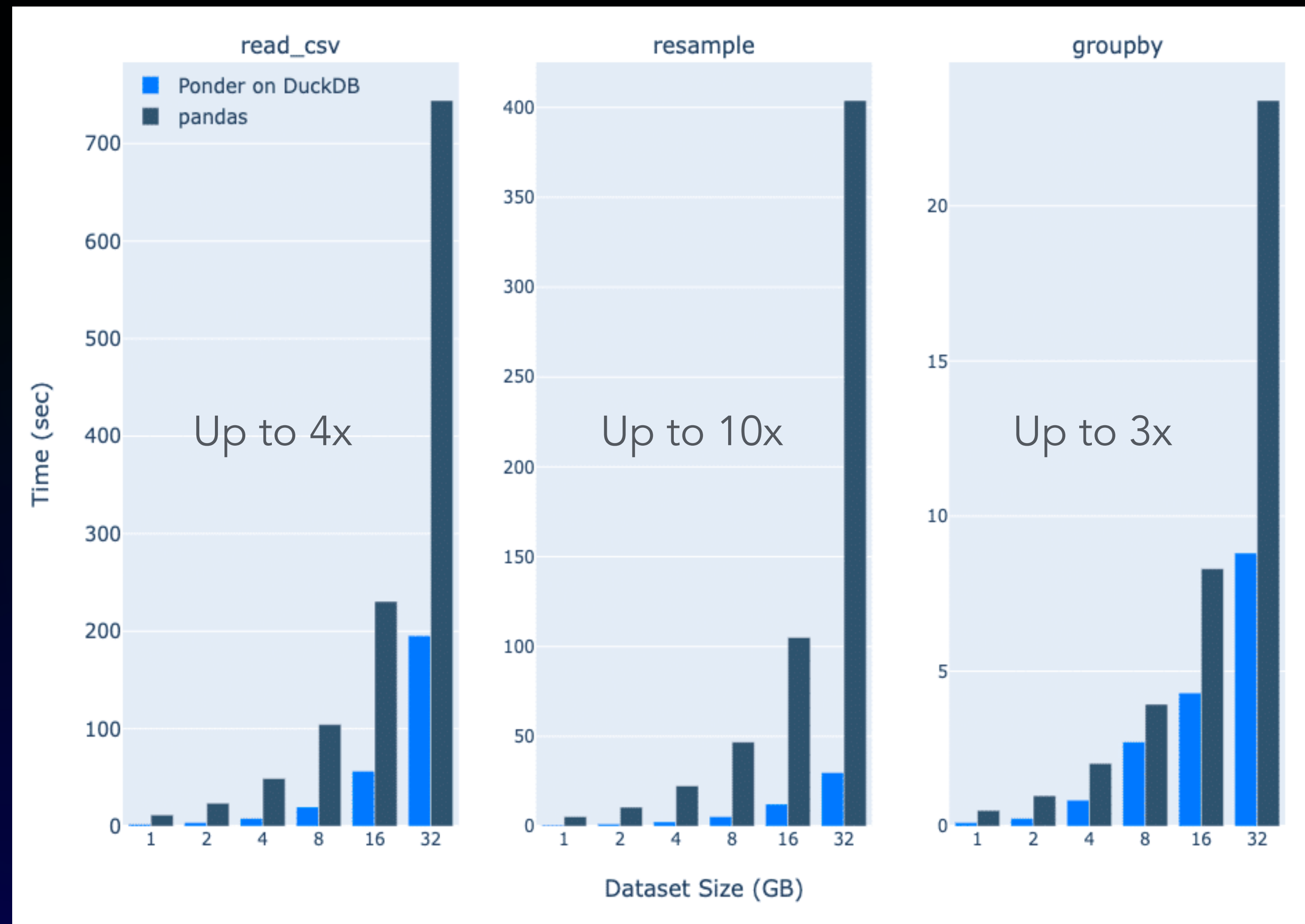


Start running pandas directly in your database

```
import ponder
ponder.init()
import modin.pandas as pd
df = pd.read_sql("DB_TABLENAME", con=db_con)

df.describe()
df.groupby("..").mean()
pd.concat([df, df]) # .. and much more! 🎨📊🔍📝
```

API-level Performance – Ponder on DuckDB vs. Pandas



Ponder on DuckDB outperforms pandas when scaling to large datasets

Two Minutes to Scalable Pandas with DuckDB!

1. Create an account:



app.ponder.io/signup

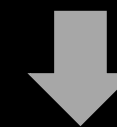
2. Install Ponder on your machine

```
$ pip install ponder
```

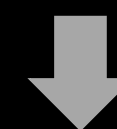
3. Log into Ponder

```
$ ponder login
```

 pandas



 PONDER



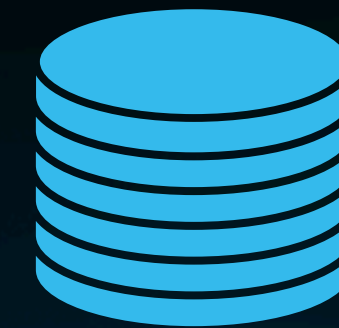
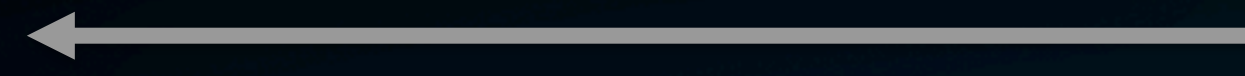
 DuckDB

Backup Slides

With  pandas

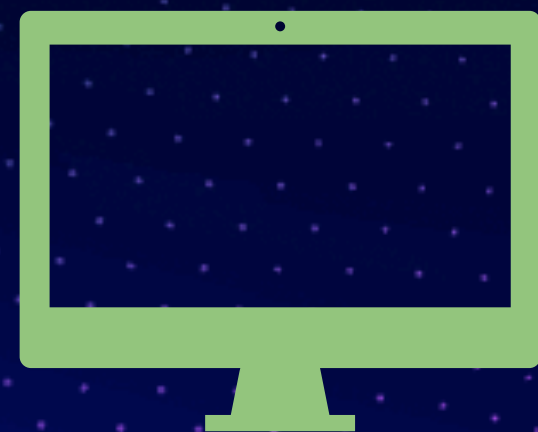


Fetch data with SQL

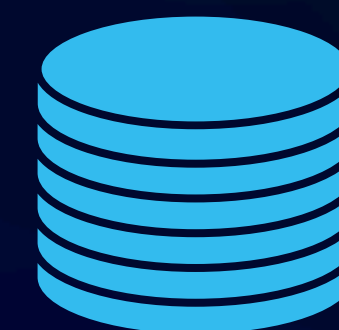


- ✗ Limited samples
- ✗ Out-of-memory errors
- ✗ Expensive I/O costs
- ✗ Un-secure access

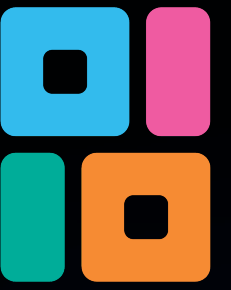
With  PONDER - Let your database do the heavy lifting



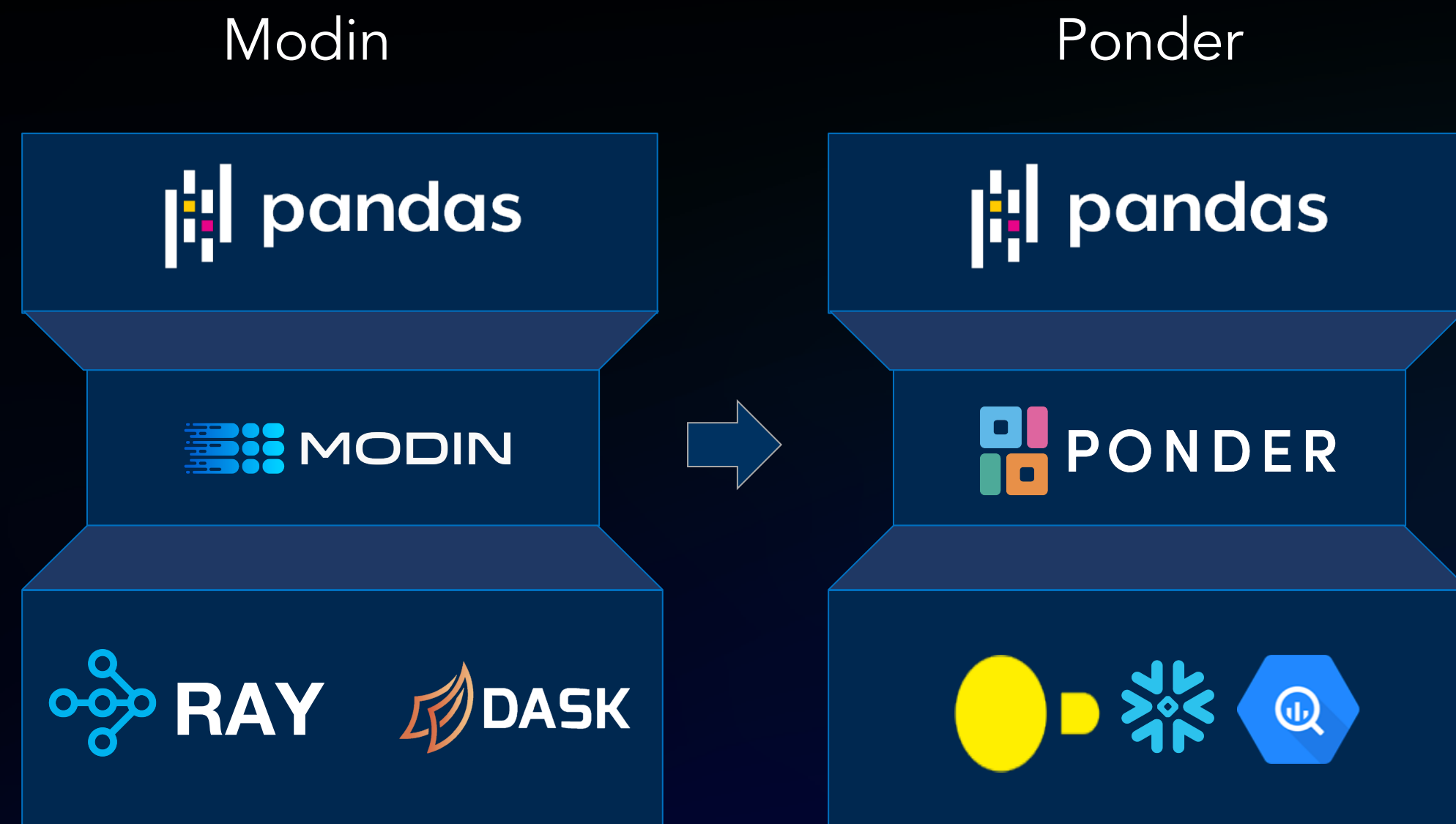
Run pandas directly



- ✓ Ready for production
- ✓ Leverage database optimizations
- ✓ No infrastructure setup required
- ✓ Securely access data where it lives



Why was this Difficult?



It is difficult to map Pandas to SQL, while supporting

- Order Semantics
- Metadata (mixed types, data/metadata equivalence)
- Incremental query construction
- Coverage of 600+ APIs (incl. ops not possible in SQL systems)

Modin instead maps to distrib. compute engines (execute anything they're told).