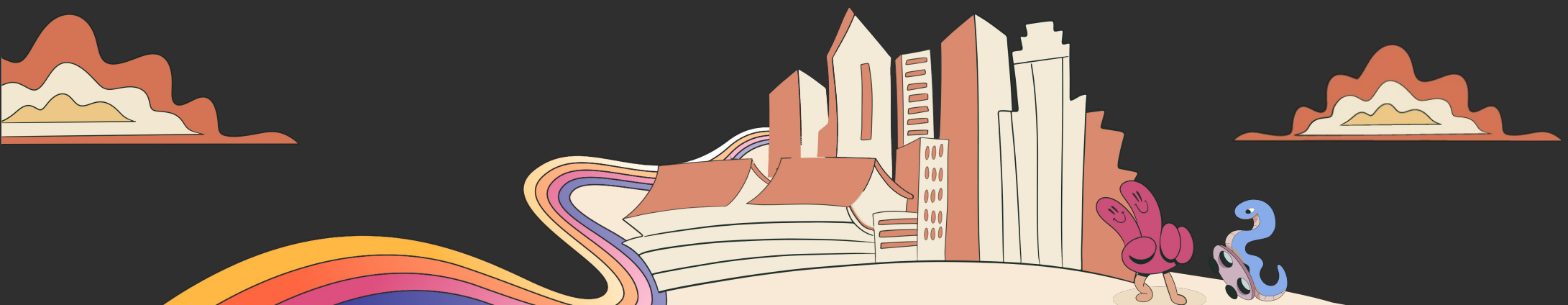
Enterprise-Ready Python for High-Performance Data Teams

Savannah (Ostrowski) Bailey, Doris Lee



PyConUS 2025

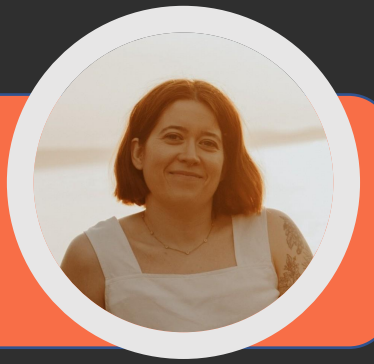
Save the slides!

Slides for this talk can be found on GitHub

savannahostrowski/pycon-2025

PyConUS 2025

Savannah (Ostrowski) Bailey



- 🐍 Python Core Developer
- 🚀 Faster CPython team
- 🧪 Jupyter Foundation Governing Board
- 💙 10+ years writing Python (3rd in-person PyCon!)
- 🛠️ Prev: Pylance/Python in VS Code, Docker, Azure Developer CLI
- 📓 Now: Product Lead for Notebooks and Python

PyConUS 2025

Doris Lee

dorisjlee

🎯 Build intelligent, easy-to-use tools for data science

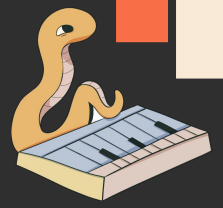
🔭 Started career in astronomy (undergrad)

🐻 CS PhD @ UC Berkeley
(Research in data viz, notebooks, Python - developed OSS projects)

🚀 Co-founder & CEO of Ponder (company behind Modin)

🤝 Acquired by Snowflake in 2023

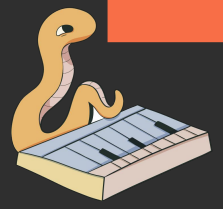
❄️ Now: Building Python Product @ Snowflake



PyConUS 2025

Agenda

- Snowflake + PyData
- Lifecycle of data to model for enterprise team
- Enterprise-ready PyData APIs
- Enterprise-grade Runtime and Environments
- Best practices for PyData at Scale



PyConUS 2025

Snowflake: Mobilize Data, Apps, and AI

PLATFORM

ANALYTICS

AI

DATA ENGINEERING

APPLICATIONS & COLLABORATION

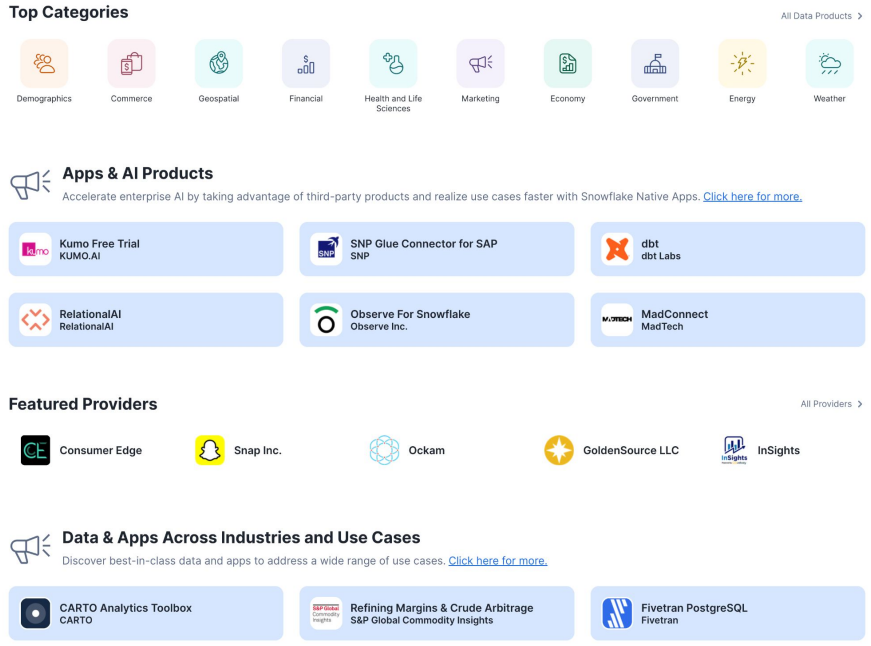
SECURITY & GOVERNANCE

COMPUTE

DATA

+

CONTENT



PyConUS 2025

Python in Snowflake

Any data app or workload, all on Snowflake

Snowflake Notebooks

Explore, analyze and visualize data using Python and SQL

Snowpark Python

Query and transform data with familiar pandas or Spark API

Streamlit in Snowflake

Turn data and AI into interactive apps with Python

PyData 🤝 Snowflake: Embracing Open Standards

Bring the tools you know and love from PyData Ecosystem

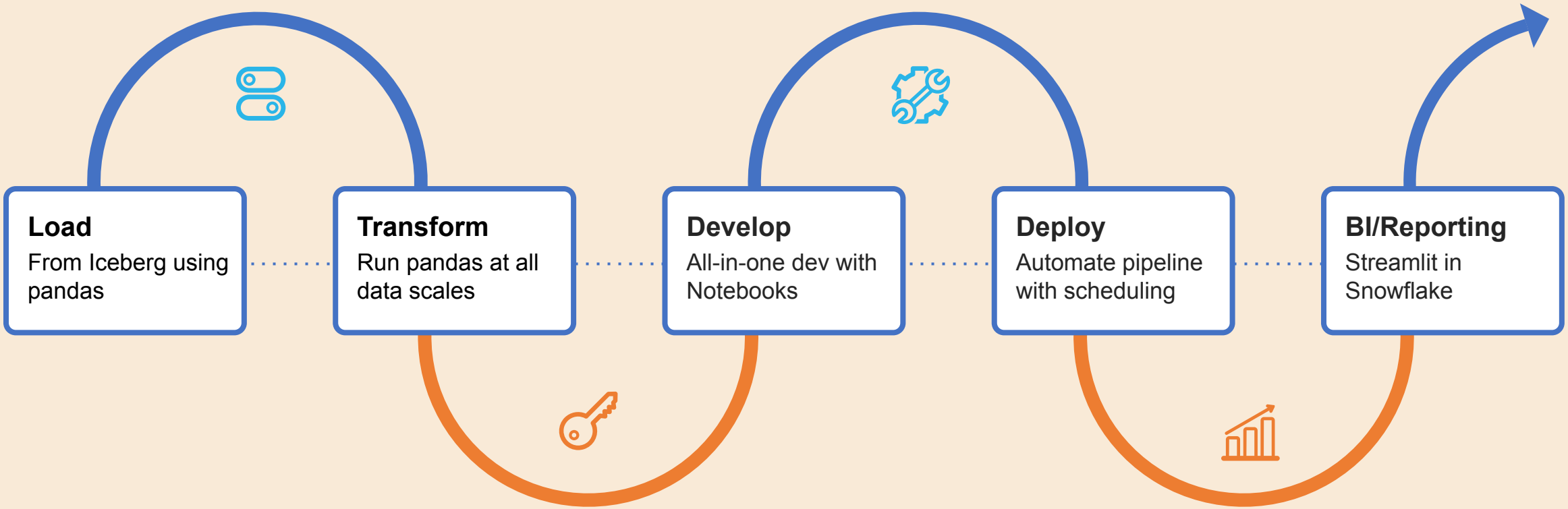
IDE — From Snowflake Notebooks or any IDE

APIs — Query and transform data with familiar APIs

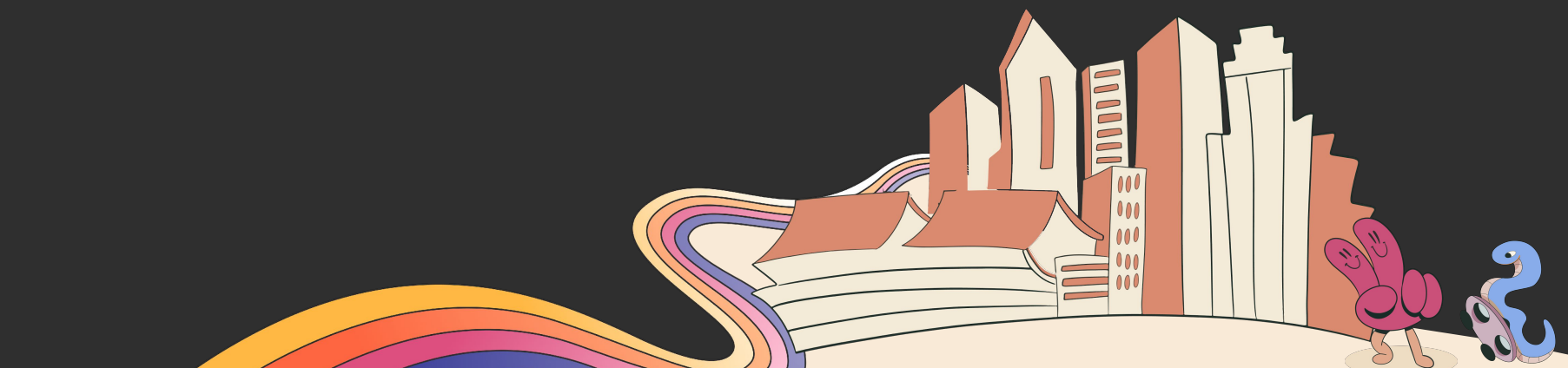
Packages — Bring your own packages and libraries

From data to model

End-to-end development for high performance data teams



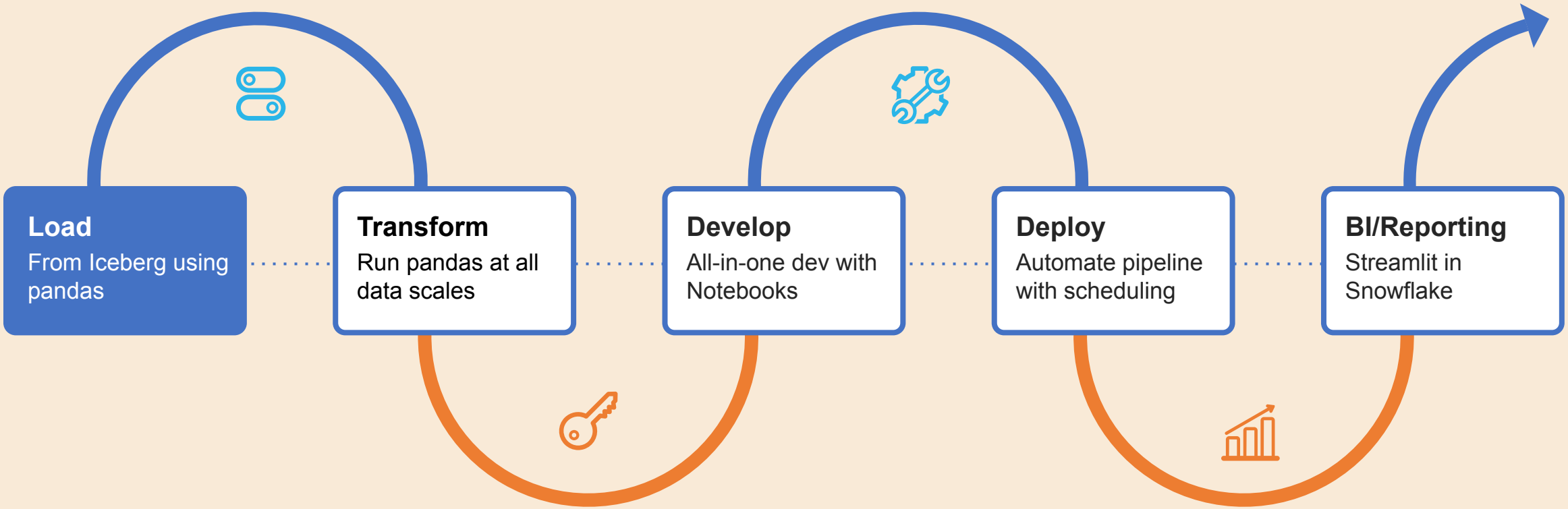
Enterprise-ready PyData APIs



PyConUS 2025

From data to model

Load data from any source to work with using pandas



What is pandas?

The de-facto Python library for data cleaning, transformation, and analysis

#1 most popular Python library
(used by 1 in 5 developers)

Flexible (600+ APIs!)

Quick Prototyping

Easy-to-use

Widely adopted

`df.describe(…)`

`df.rename(…)`

`pd.concat(…)`

`df.dropna(…)`

`df.groupby(…)`

`pd.join(…)`

`pd.merge(…)`

`df.pivot(…)`

PyCon US 2025

Bringing best of pandas to Snowflake

PANDAS ON SNOWFLAKE

WHAT IS IT

Simple, unified interface for working with Snowflake data using familiar pandas API

WHY USE IT

- Easily work with Snowflake data and bring in data from files
- Develop pandas pipeline at all data scales from prototype to production
- Powerful Snowflake analytics through convenient pandas semantics for flexible transformation

```python
import modin.pandas as pd
import snowflake.snowpark.modin.plugin
# Load data from a Snowflake table (Supports Iceberg, views, etc.)
df = pd.read_snowflake('orders')

# Convert date column to datetime type for resampling
df['date'] = pd.to_datetime(df['date'])

# Classify text using Cortex LLM functions
from snowflake.cortex import ClassifyText
df['region'] = df["location"].apply(ClassifyText, /
               categories=['North America', 'Europe', 'Asia'])

# Resample to daily granularity and fill missing days with zero
daily_sales = df.resample('D').sum()
daily_sales_filled = daily_sales.fillna(0)

# Save result to a dynamic table in Snowflake
df.to_dynamic_table('processed_orders')
```

Loading data with pandas on Snowflake

Seamlessly interact with Snowflake objects

```
# Supports Tables, views, Iceberg, Dynamic Tables, etc.
df = pd.read_snowflake("MY_TABLE")
```

Get Code Example

https://github.com/Snowflake-Labs/snowflake-python-recipes/blob/main/Read%20from%20Table/Reading%20from%20Table.ipynb

Load files from local or stage paths across a range of formats

```
# Reading CSV from S3 and other remote source
df = pd.read_csv("s3://mys3bucket/data.csv")
# Reading Parquet from stage file
df = pd.read_parquet("@MY_STAGE/data.parquet")
```

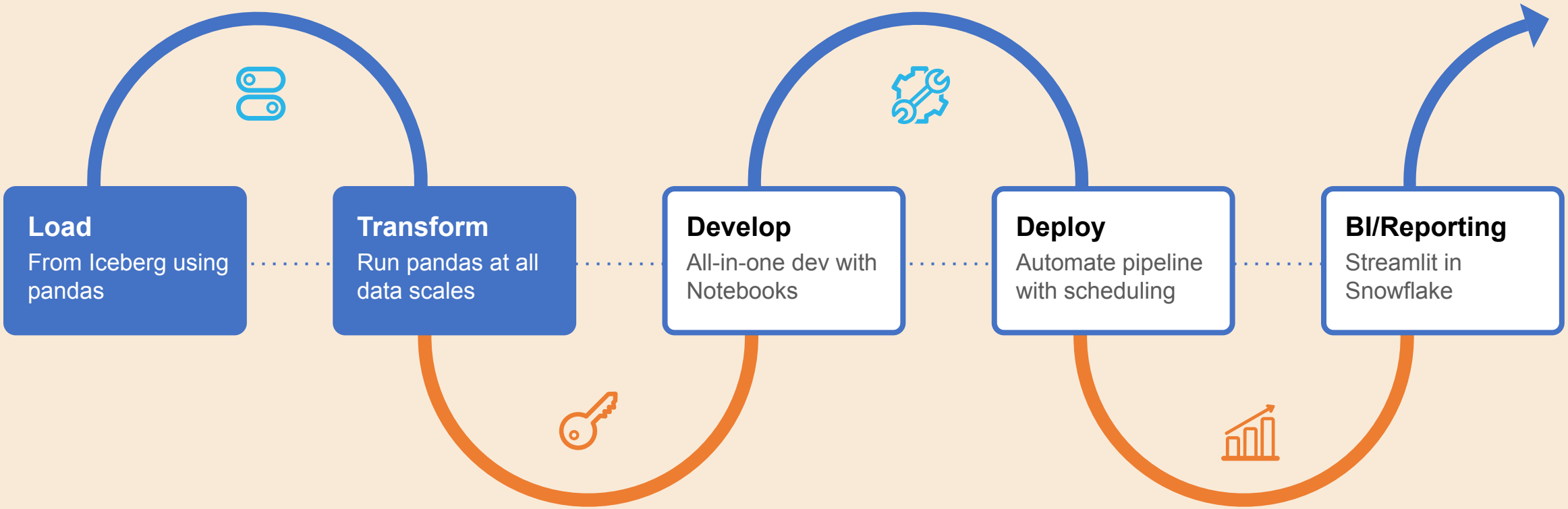
Get Code Example

https://github.com/Snowflake-Labs/snowflake-python-recipes/blob/main/Load%20files%20to%20Snowflake/load_files_snowflake.ipynb

PyCon US 2025

From ingest to transformation

End-to-end development for high performance data teams

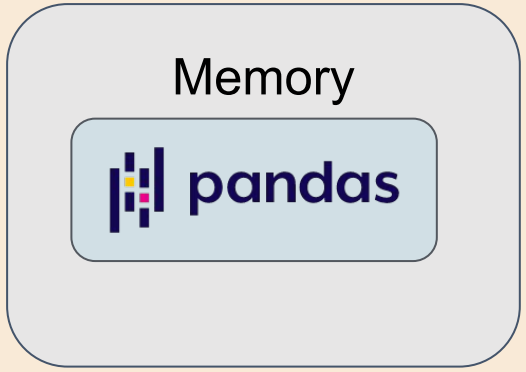


Challenges of working with enterprise-scale data

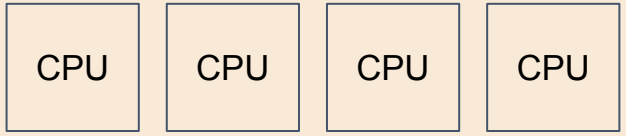
6+ months

Time spent to bring pandas workloads to production

How pandas works?

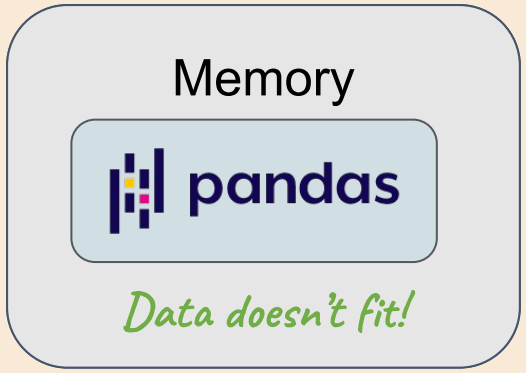


pandas operates on data in-memory



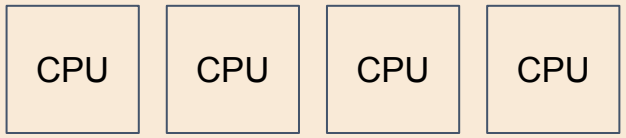
pandas is single threaded

Enter Enterprise-scale data…



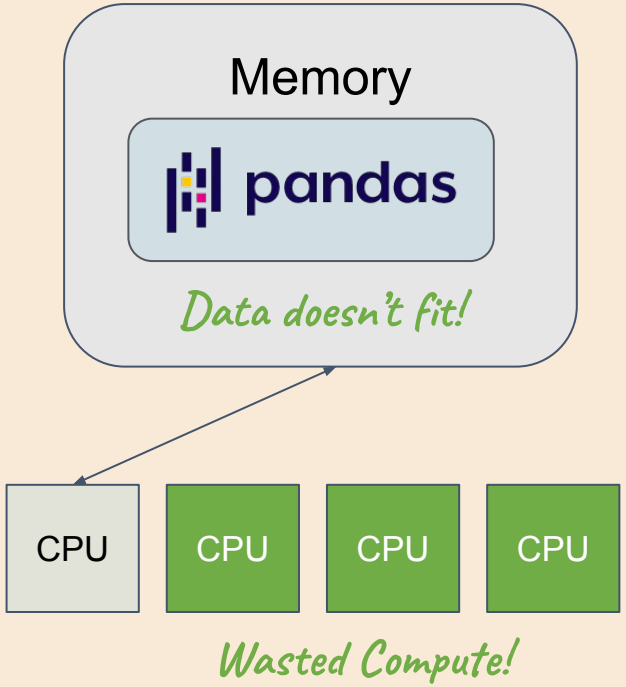
pandas operates on data in-memory

🛑 **Out of memory errors!**



pandas is single threaded

Enter Enterprise-scale data…



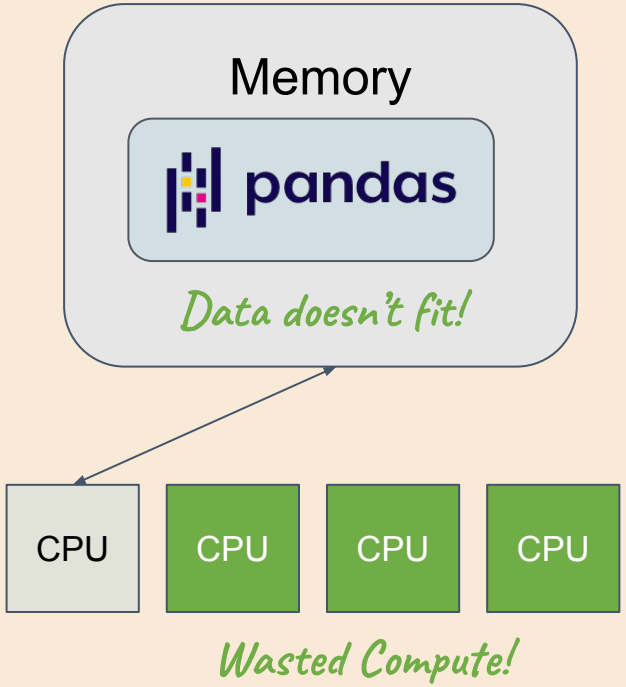
pandas operates on data in-memory

- **Out of memory errors!**

pandas is single threaded

- **Slow – No distributed processing!**

Enter Enterprise-scale data…



pandas operates on data in-memory

- **Out of memory errors!**

pandas is single threaded

- **Slow – No distributed processing!**

… pandas doesn't scale (even on ~ tens GB/millions rows)

What this means for data teams

Costly Re-translation

Difficult to experiment & debug

Loss of productivity

MODIN

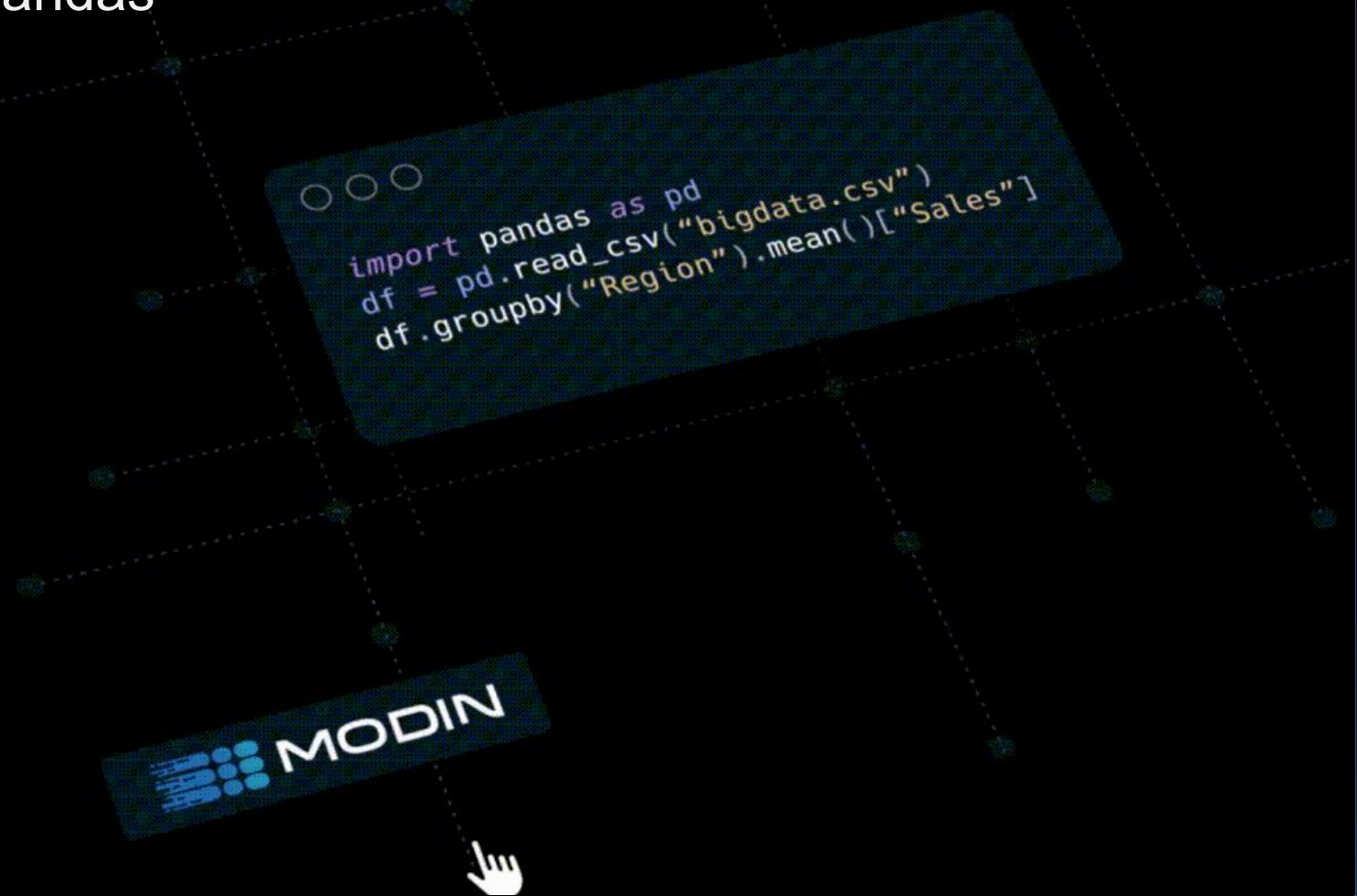
Scalable “drop-in” replacement for pandas

modin_project/modin ☆ Star 10k+

🔥 **1.3M+ monthly** downloads
🤝 **100+** contributors
🚀 Used by **1K+ organizations**

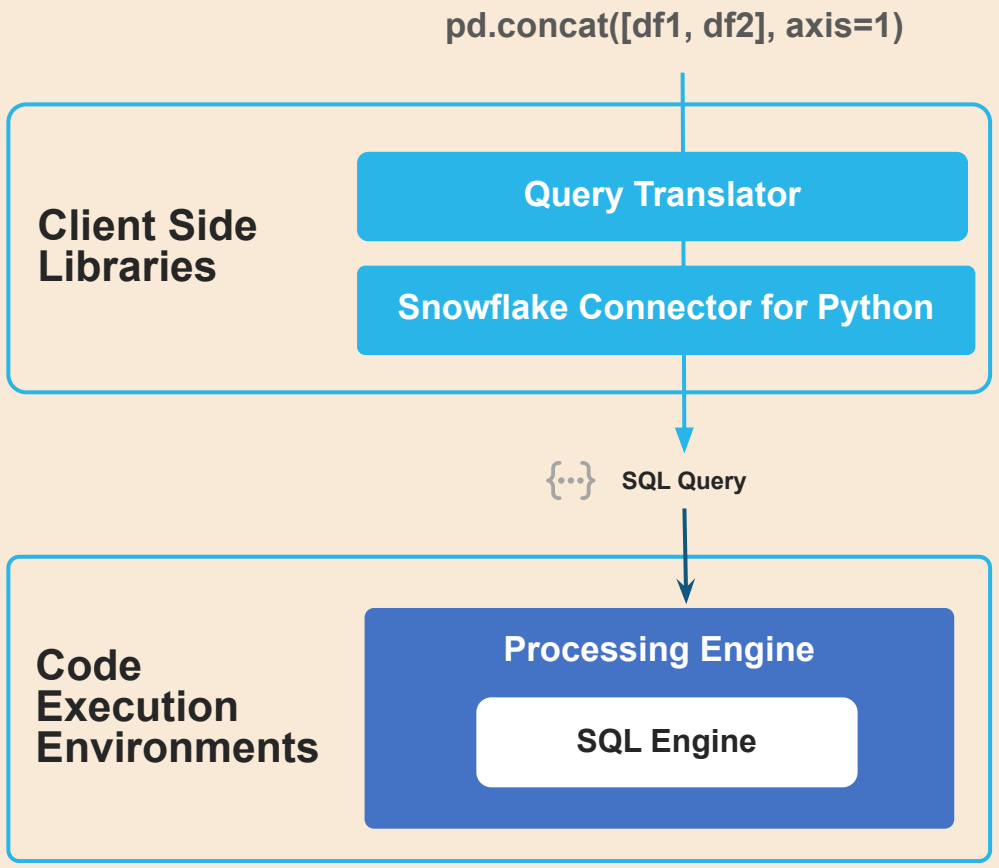
Runs on scalable OSS backends:

RAY DASK Snowpark



Distributed Pandas at Scale in Snowflake

Automatic translation from pandas to SQL - **no code rewrite required**

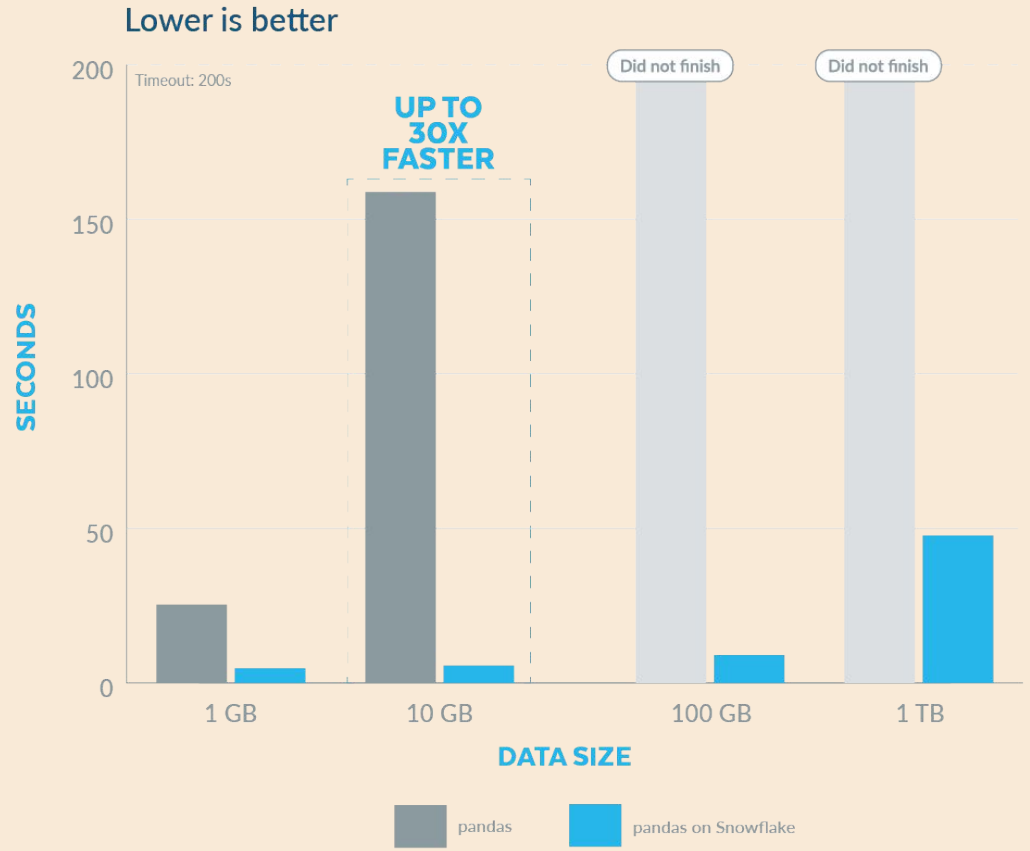


Seamless Prototype to Production
No need to manage or tune compute infrastructure

Zero Data Movement
Compute happens directly in Snowflake securely

Write with pandas
Leverage familiar API. No rewrites necessary.

Run pandas on 1 TB+ enterprise data



Benchmark testing across different data sizes shows how pandas on Snowflake can scale up to 1TB and is up to 30x faster than pandas at the 10GB scale.

30X+ faster performance than pandas

- Address painful out of memory issues to scale pandas
- Secure access within Snowflake removes sensitive data risks on local machines
- Accelerate prototype to production and scale without tuning or managing infrastructure

Embracing Open Standards

Integration with PyData Libraries through Dataframe Interchange Protocol

- pandas is often used with third-party libraries for visualization, ML.
- Added support for dataframe interchange protocol
- Increase interoperability with other PyData libraries

Get Started Today!

Installation

Use with Snowflake Notebook or in your own IDE

```
$ pip install "snowflake-snowpark-python[modin]"
```

```
import modin.pandas as pd
import snowflake.snowpark.modin.plugin
```

Resources

Developer Guide

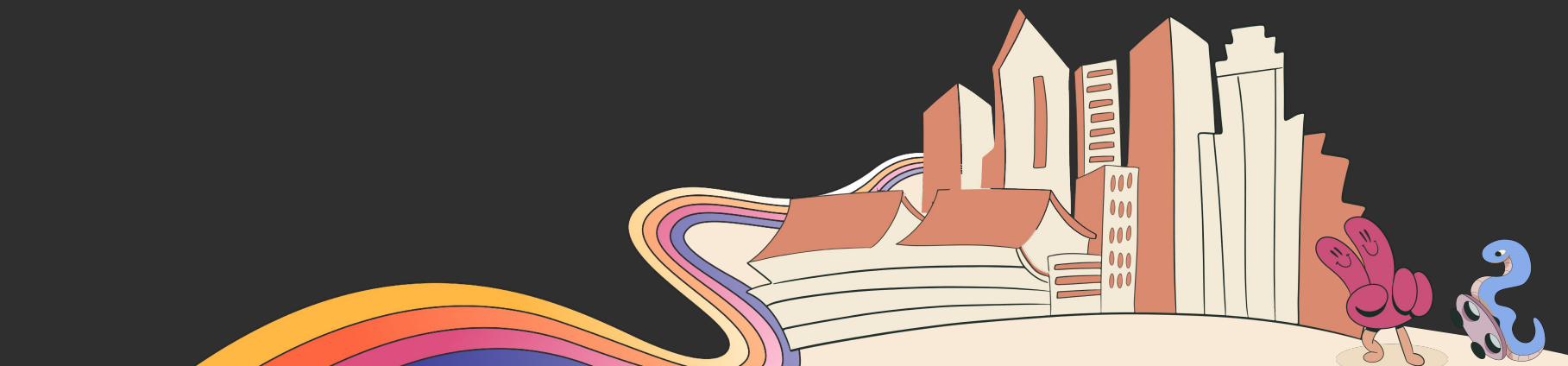
https://docs.snowflake.com/en/developer-guide/snowpark/python/pandas-on-snowflake

Quickstart

https://quickstarts.snowflake.com/guide/getting_started_with_pandas_on_snowflake

PyCon US 2025

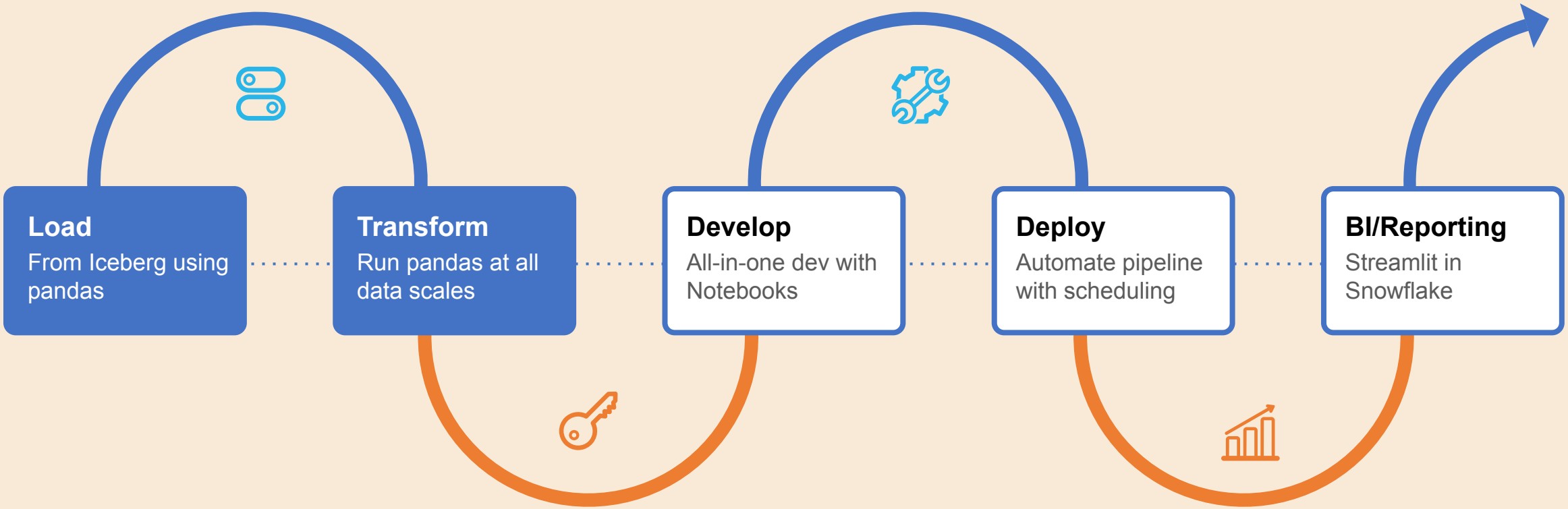
Enterprise-grade runtimes and environments



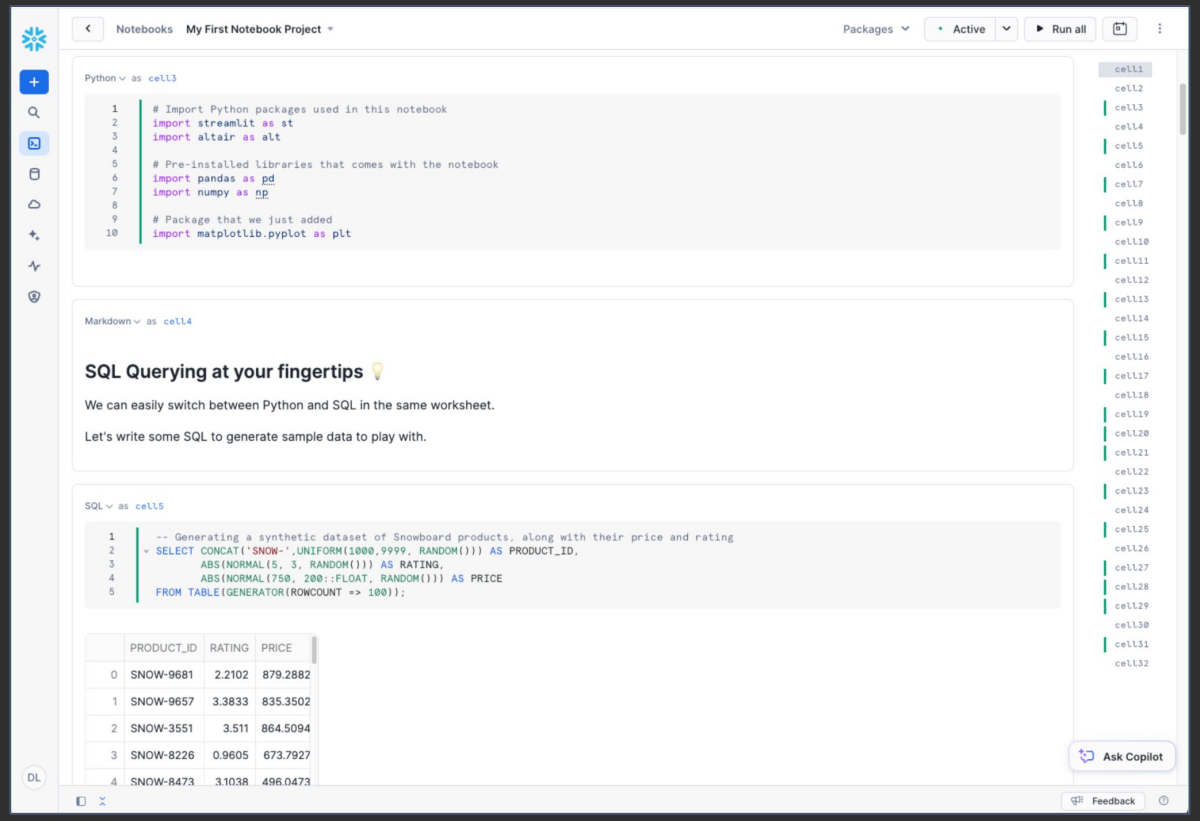
PyConUS 2025

From exploration to model

End-to-end development for high performance data teams



Snowflake Notebooks



A unified cell-based, interactive development surface that makes it faster and easier than ever to leverage your Snowflake data in Data Science, ML, and Data Engineering workflows

Instant access to data from your Notebook

Focus on code, not configuring access to your data

Seamless interop between languages

Blend SQL and Python with cell and variable referencing

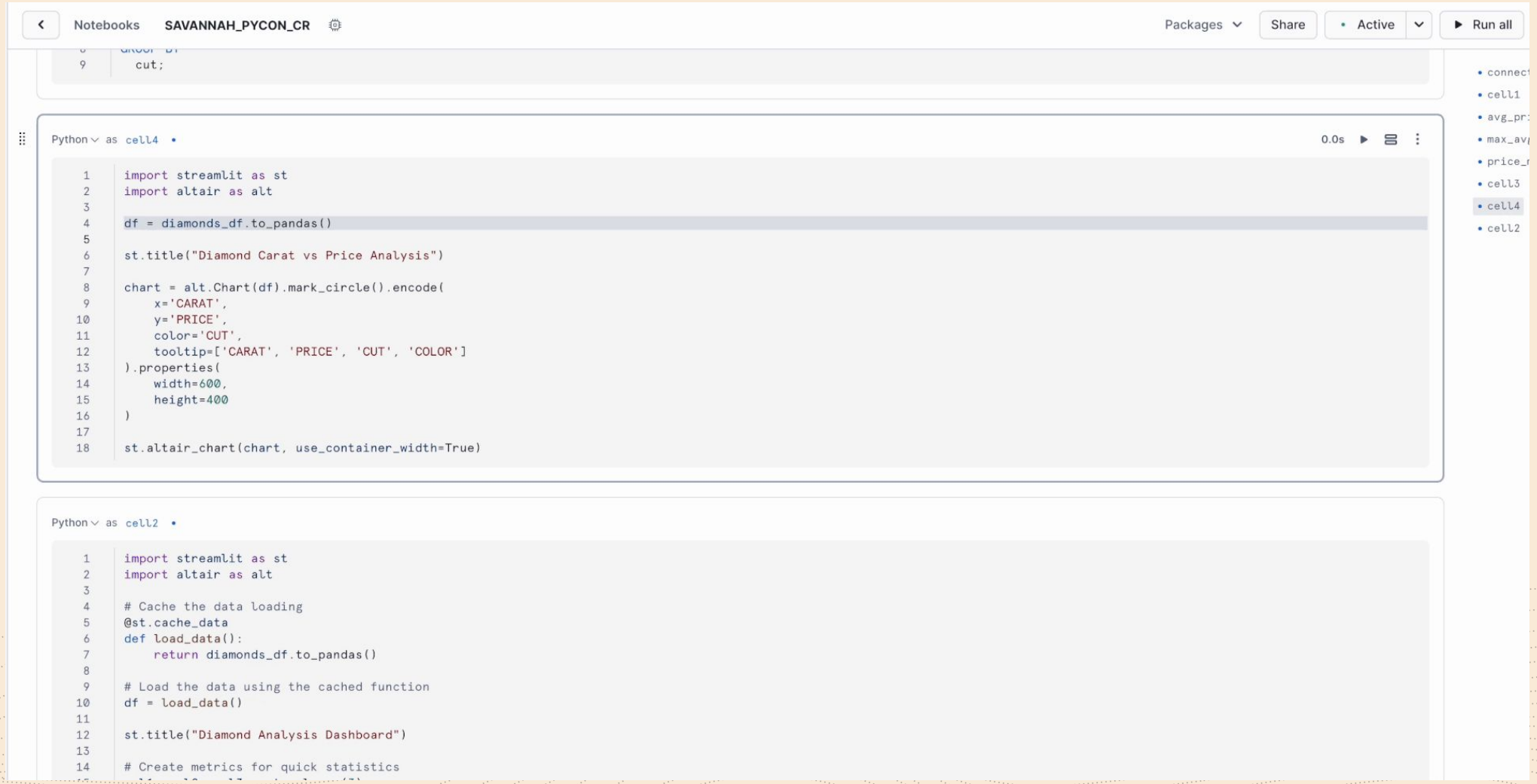
Code Example

https://github.com/Snowflake-Labs/snowflake-python-recipes/blob/main/Python%20SQL%20cells/python-sql-cells.ipynb

PyCon US 2025

Interactive data visualization in < 100 lines of code

Embed interactive Streamlit apps directly in your Notebook for deeper insights



Code Example

https://github.com/Snowflake-Labs/snowflake-python-recipes/blob/main/Data%20Visualization%20with%20Streamlit/data_viz_with_streamlit.ipynb

PyCon US 2025

Challenges in production ML development

Cost vs performance

Meeting the performance and quality standards while attempting to keep **costs under control** is challenging

Scalability

With complex model schemas and large datasets, training models require **sophisticated infrastructure** capable of that scale

Governance

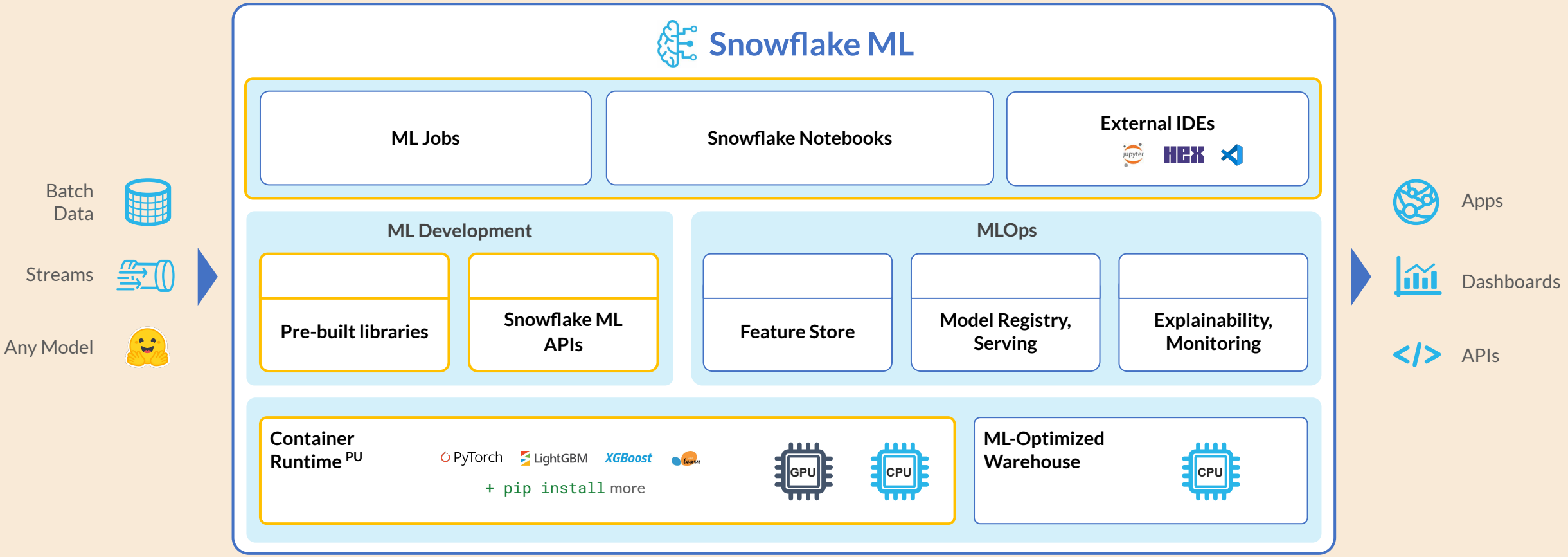
Maintainability and reproducibility **requires consistency in build environments and effective isolation** – often a challenging DevOps problem

Access to GPUs

Deep learning and LLM related tasks require processing on GPUs – **often hard to access without administrative setup or DevOps knowledge**

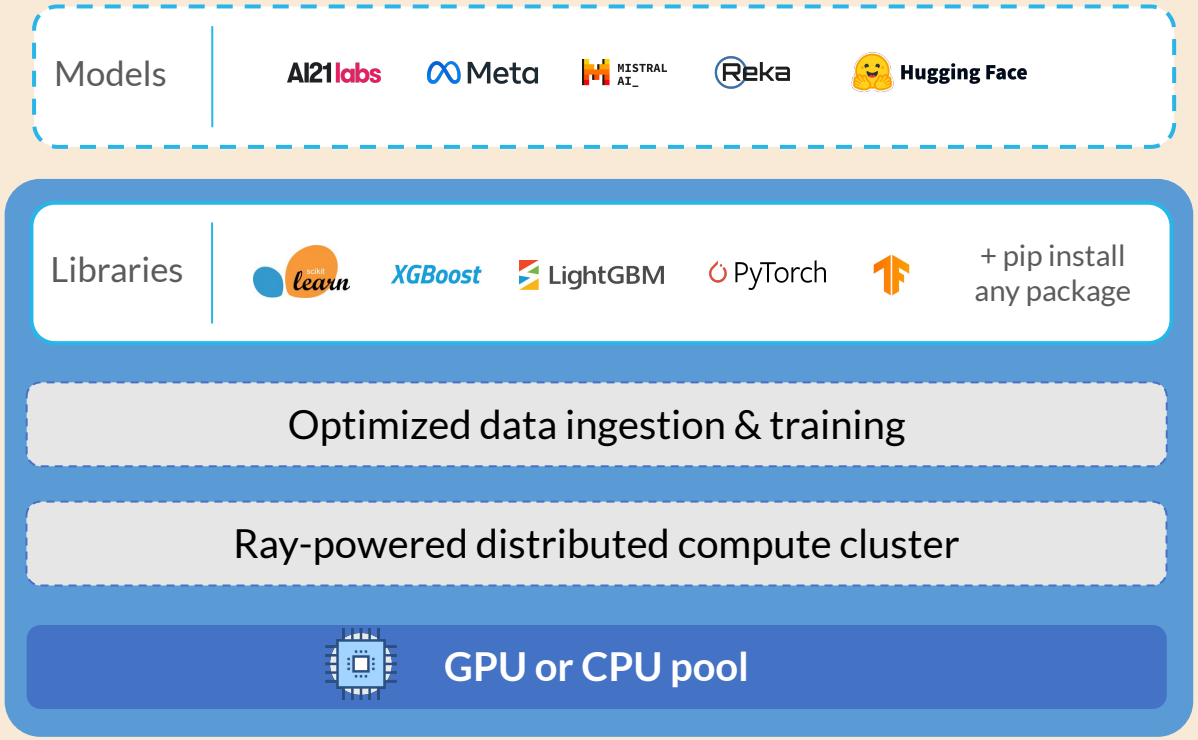
Snowflake ML

Integrated set of capabilities for end-to-end ML in Snowflake



PU: Public Preview

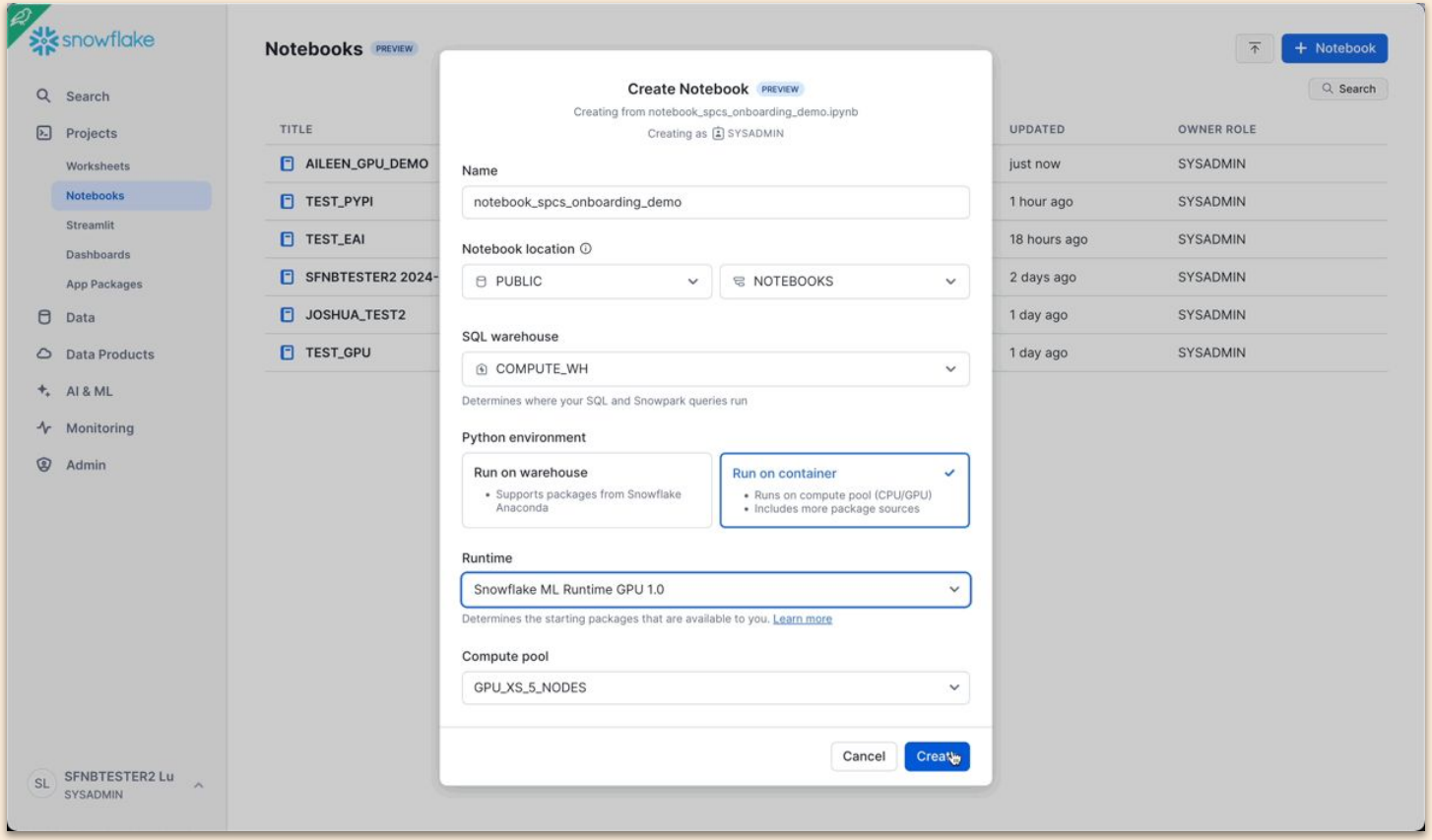
Container Runtime for ML



- Pre-built container environment with **latest ML packages** already installed
- Data connector APIs for **optimized data ingestion**
- Optimized and **scalable training APIs** for frameworks like XGBoost, LightGBM, PyTorch
- **Distributed hyperparameter optimization** using Bayesian strategy
- Ray-powered compute cluster for efficient multi-core and multi-node **distributed training**
- Easy access to **CPU and GPU** compute pools

Snowflake Notebooks on Container Runtime

- Notebooks that run on a fully managed container environment with access to CPUs and GPUs, optimized data loading from Snowflake, automatic lineage capture and Model Registry integration
- Flexibility to run ML workloads at scale without data movement
- Leverage pre-installed ML packaged or pip to install any custom package



Optimized ML APIs

Data Loading with DataConnector API

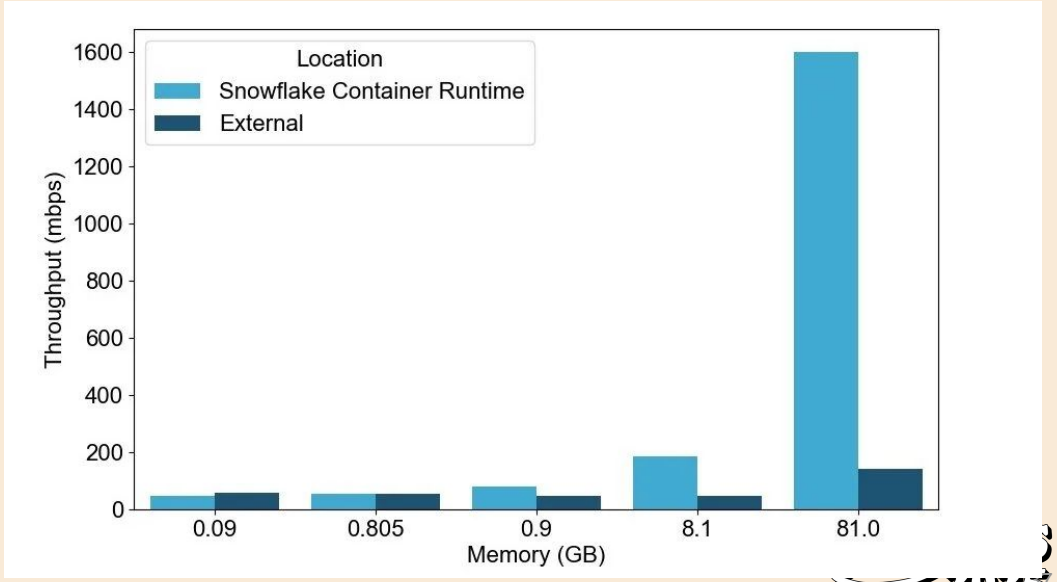
With DataConnector API

81 GB of data into pandas DataFrame in 50 seconds vs. ~9.5 minutes with OSS equivalent outside Snowflake

```
Python as cell19

1   from snowflake.ml.data.data_connector import DataConnector
2
3   # Retrieve data from a snowflake table
4   table_name = 'MLR_BENCHMARK_DB.PUBLIC.REGRESSION_DATA_12GB' # 12GB table
5   snowpark_df = session.table(table_name)
6
7   # Materiazed it into a pandas dataframe using DataConnector
8   t0 = time.time()
9   pandas_df = DataConnector.from_dataframe(snowpark_df).to_pandas()
10  duration = time.time() - t0
11  print(f'Materialization of 12GB data uses {duration:.2f} secs')

Materialization of 12GB data uses 7.36 secs
```



Optimized ML APIs

Distributed Training APIs

```python
from snowflake.ml.modeling.distributors.xgboost import XGBEstimator, XGBScalingConfig
from snowflake.ml.data.data_connector import DataConnector

table_name = 'MLR_BENCHMARK_DB.PUBLIC.REGRESSION_DATA_12GB' # 12GB table
snowpark_df = session.table(table_name)
data = DataConnector.from_dataframe(snowpark_df)
input_cols = ["FEATURE1", "FEATURE2", "FEATURE3", "FEATURE4", "FEATURE5", "FEATURE6", "FEATURE7"]
label_col = 'TARGET'

# Leverage GPU, and let system figure out the rest by passing in -1
scaling_config = XGBScalingConfig(use_gpu=True, num_workers=-1, num_cpu_per_worker=-1)
estimator = XGBEstimator(n_estimators=10, scaling_config=scaling_config)

model = estimator.fit(data, input_cols=input_cols, label_col=label_col)
```

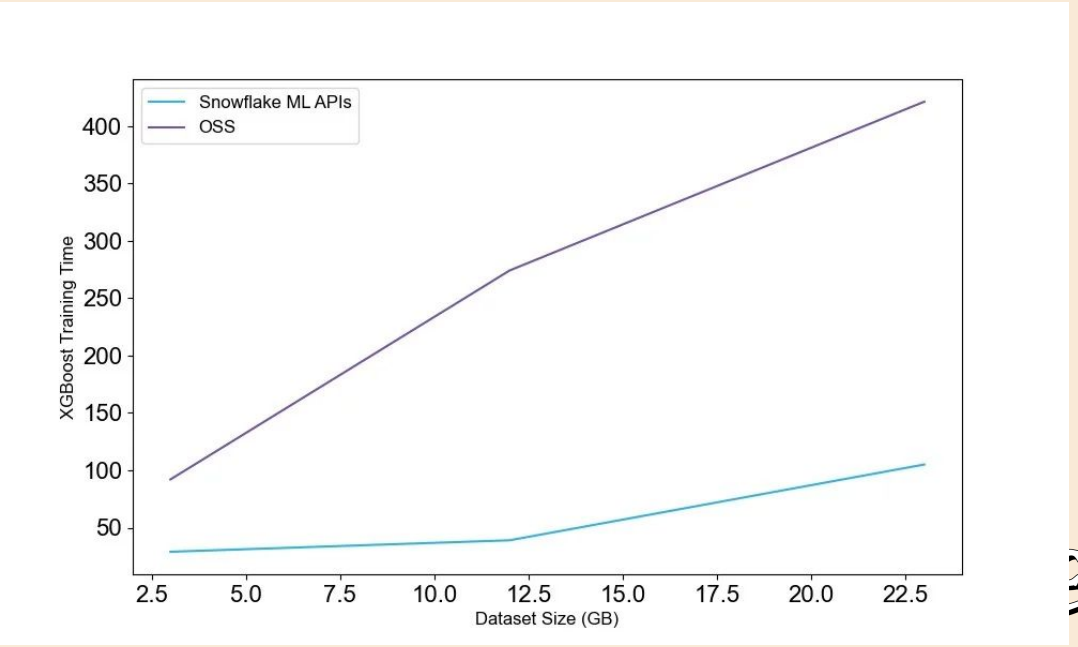
```
2024-09-10 23:24:23,436 INFO tune.py:622 -- [output] This will use the new output engine with verbosity

Training started without custom configuration.

[36m(SplitCoordinator pid=6502) [0m Starting execution of Dataset. Full log is in /tmp/ray/session_2024
```

With Snowflake ML - XGBoost

3-7x speed up compared to open source XGBoost by utilizing all four available GPUs

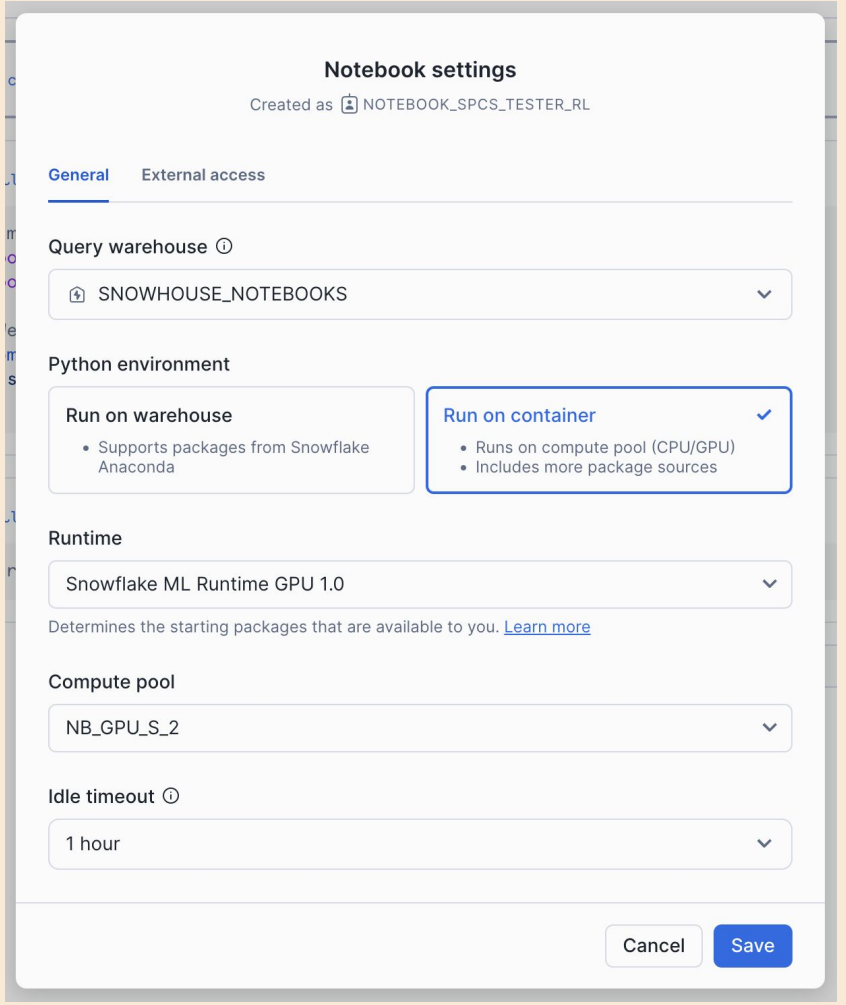


Distributed APIs for PyTorch, XGBoost & LightGBM available!

Managed Access to GPUs

With admin-led governance

- ✓ Admin configured GPU pools easily accessible via Notebook config
- ✓ Easily scale up and down *during a NB session* to only use resources as needed
- ✓ Snowflake ML APIs configured to automatically detect and utilize all available GPUs



Containerized ML Environment

CPU and GPU-specific runtime images

- ✓ Comprehensive list of pre-installed packages curated for ML development
- ✓ Flexibility to supplement with any open source package with `pip`
- ✓ Pre-configured CUDA libraries to easily leverage GPUs

Full list for GPU v1 image

This table is a full list of Python packages pre-installed on the GPU v1 image:

Package	Version	Package	Version
Babel	2.11.0	Bottleneck	1.3.7
Brotli	1.0.9	Brotli	1.0.9
GitPython	3.1.41	Jinja2	3.1.3
MarkupSafe	2.1.3	PyJWT	2.4.0
PySocks	1.7.1	PySocks	1.7.1
PyYAML	6.0.1	Pygments	2.15.1
Pympler	0.9	Send2Trash	1.8.2
absl-py	1.4.0	accelerate	0.28.0
aiobotocore	2.7.0	aiohttp	3.9.1
aiohttp	3.9.3	aiohttp-cors	0.7.0
aioitertools	0.7.1	aioprometheus	23.12.0
aiosignal	1.2.0	aiosignal	1.3.1
altair	4.2.2	annotated-types	0.6.0

Run your Notebook on a schedule

Keep your analysis or model up to date using scheduling

Documentation

https://docs.snowflake.com/en/user-guide/ui-snowsight/notebooks-schedule

PyCon US 2025

Easily bring your model into Streamlit for reporting

Share your work with stakeholders

Code Example

https://github.com/Snowflake-Labs/snowflake-python-recipes/tree/main/Snowflake%20ML%20Model%20in%20Streamlit

PyCon US 2025

Learn about Snowflake Notebooks for ML

Documentation

https://docs.snowflake.com/developer-guide/snowflake-ml/notebooks-on-spcs

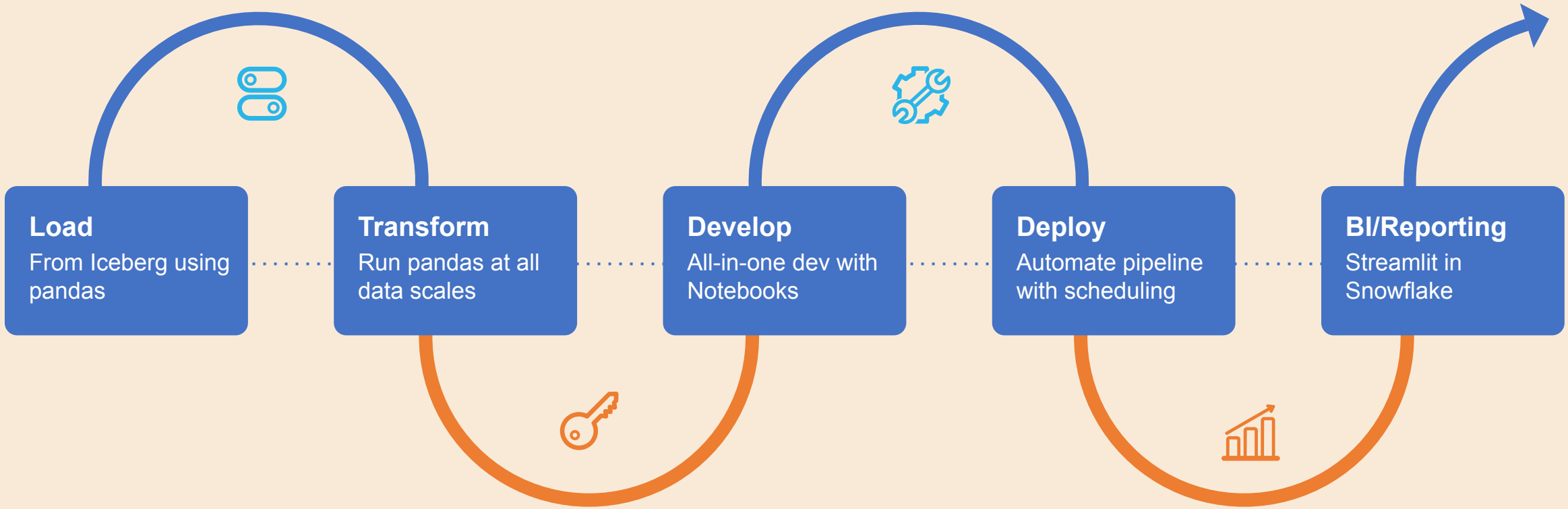
Container Runtime + Notebooks Quickstart

https://quickstarts.snowflake.com/guide/notebook-container-runtime/

PyCon US 2025

From data to model to insights

End-to-end development for high performance data teams



Snowflake 💙 PyData

Easy

Built for developer productivity. Streamline workflow practices across data teams.

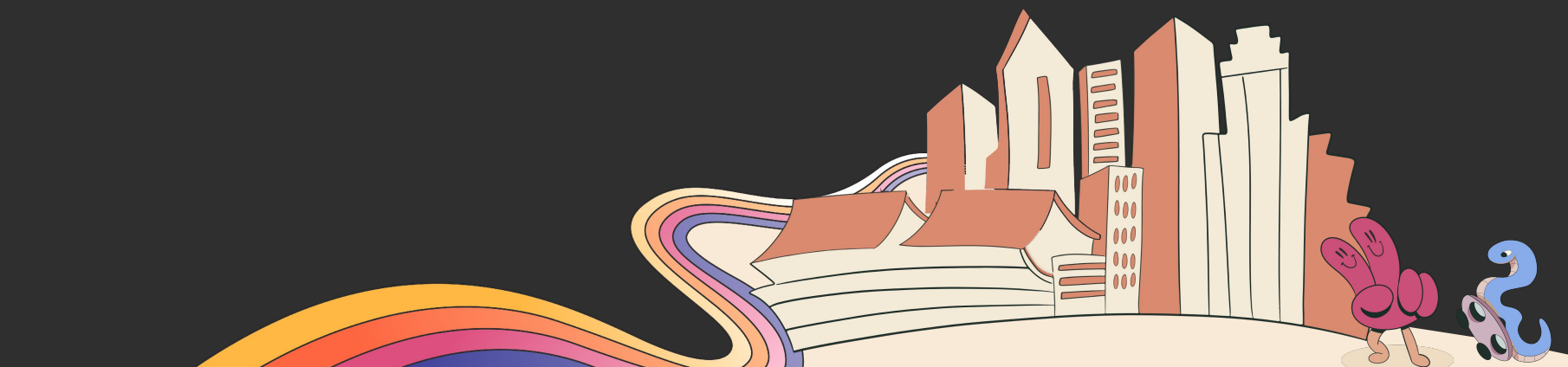
Open

Embrace open formats and standards. Bring your familiar libraries and frameworks.

Trusted

Interoperability of data, across clouds, with universal governance at scale

Thank you!



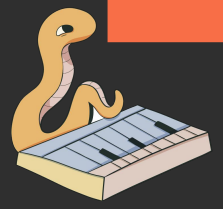
PyConUS 2025

Snowflake Python Recipes

Collection of code examples, snippets, best practices for Python in Snowflake

TRY IT ON YOUR OWN TODAY!

Snowflake-Labs/snowflake-python-recipes

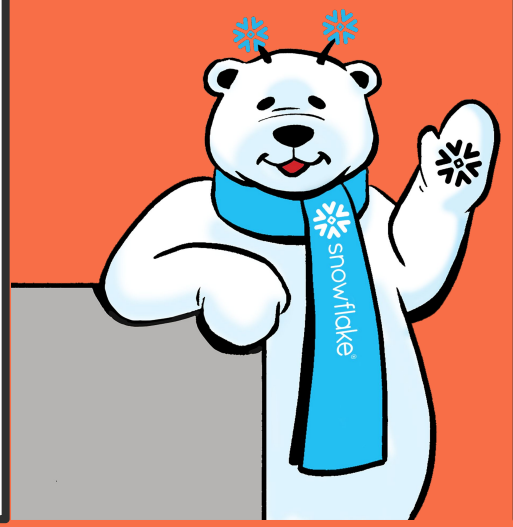


PyConUS 2025

We'd love to hear from you!

Fill out this 5 minute survey to share your experience

snowflake.co1.qualtrics.com/jfe/form/SV_036qhfYM7YwEK7s



Come by Booth 309 for SWAG

PyConUS 2025