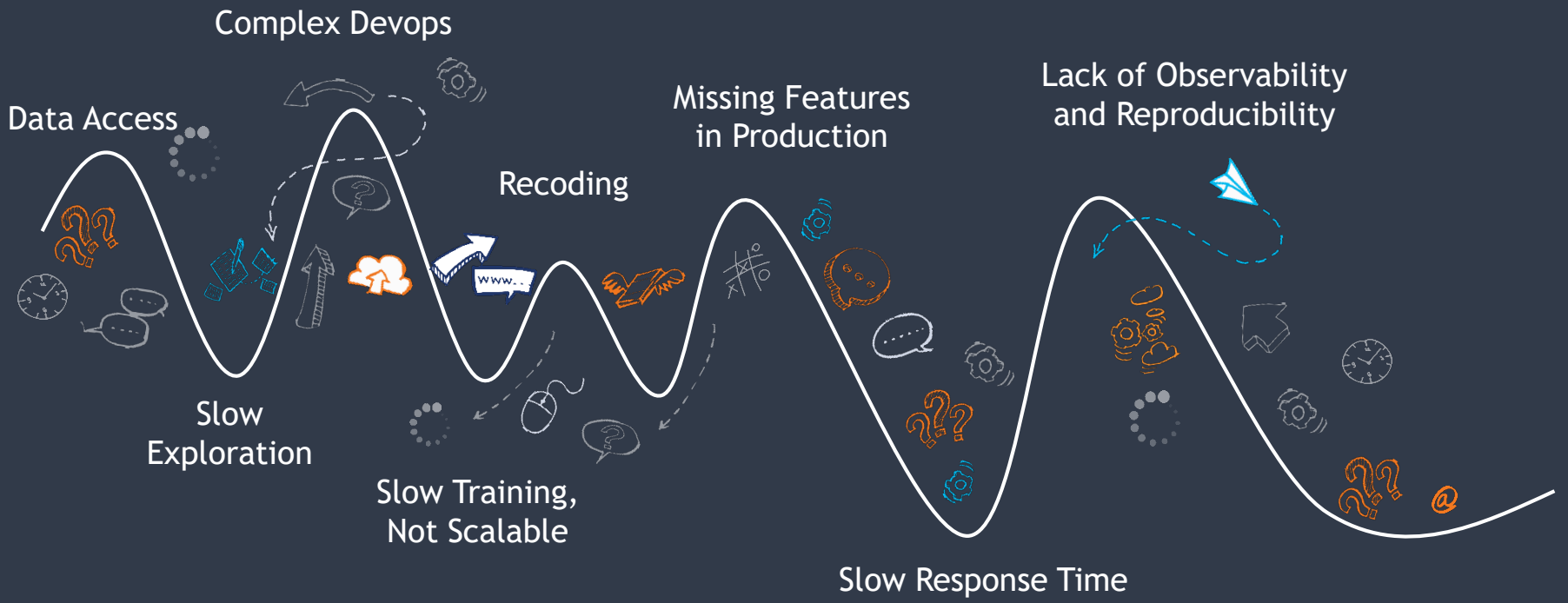
iguazio

Delivering GPU as a Service

Yaron Haviv, CTO, Iguazio

One Year From Data Science in Production

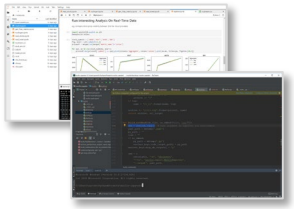


iguazio

Code/Model Development Is Just The FIRST Step

Every piece of code, data science algorithm, or data processing task must be built for production

Develop/Experiment



Package

- Dependencies
- Parameters
- Run scripts
- Build

Scale-out

- Load-balance
- Data partitions
- Model distribution
- Hyper params

Tune

- Parallelism
- GPU support
- Query tuning
- Caching

Instrument

- Monitoring
- Logging
- Versioning
- Security

Automate

- CI/CD
- Workflows
- Rolling upgrades
- A/B testing

Weeks with one data scientist

Months with a large team of developers, scientists, data engineers and DevOps

GPUs Accelerate Many Data-Science Workload

cudf

In [7]:
```
%%timeit

# Read file
gdf = cudf.read_json(benchmark_file, lines=True)

# Perform aggregation
ggdf = gdf.groupby(['company']).\
            agg({k: ['min', 'max', 'mean'] for k in metric_names})

# Get N Largest (From original df)
raw_nlargest = gdf.nlargest(nlargest, 'cpu_utilization')
```

```
1.44 s ± 23.1 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

Pandas

In [8]:
```
%%timeit

# Read file
pdf = pd.read_json(benchmark_file, lines=True)

# Perform aggregation
gpdf = pdf.groupby(['company']).\
            agg({k: ['min', 'max', 'mean'] for k in metric_names})

# Get N Largest (From original df)
raw_nlargest = pdf.nlargest(nlargest, 'cpu_utilization')
```

```
43.4 s ± 627 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
```

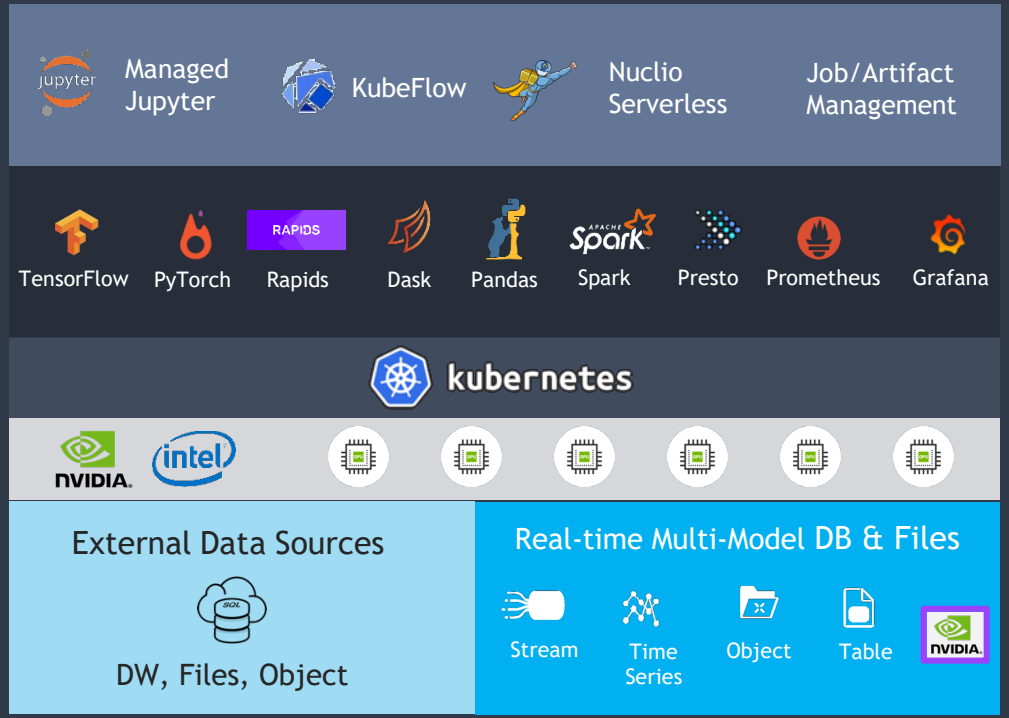
30x faster data analytics

BUT we don't want to pay for their idle time

See: https://towardsdatascience.com/python-pandas-at-extreme-performance-912912b1047c

iguazio

Building a Cloud-Native Data Science and ML Platform



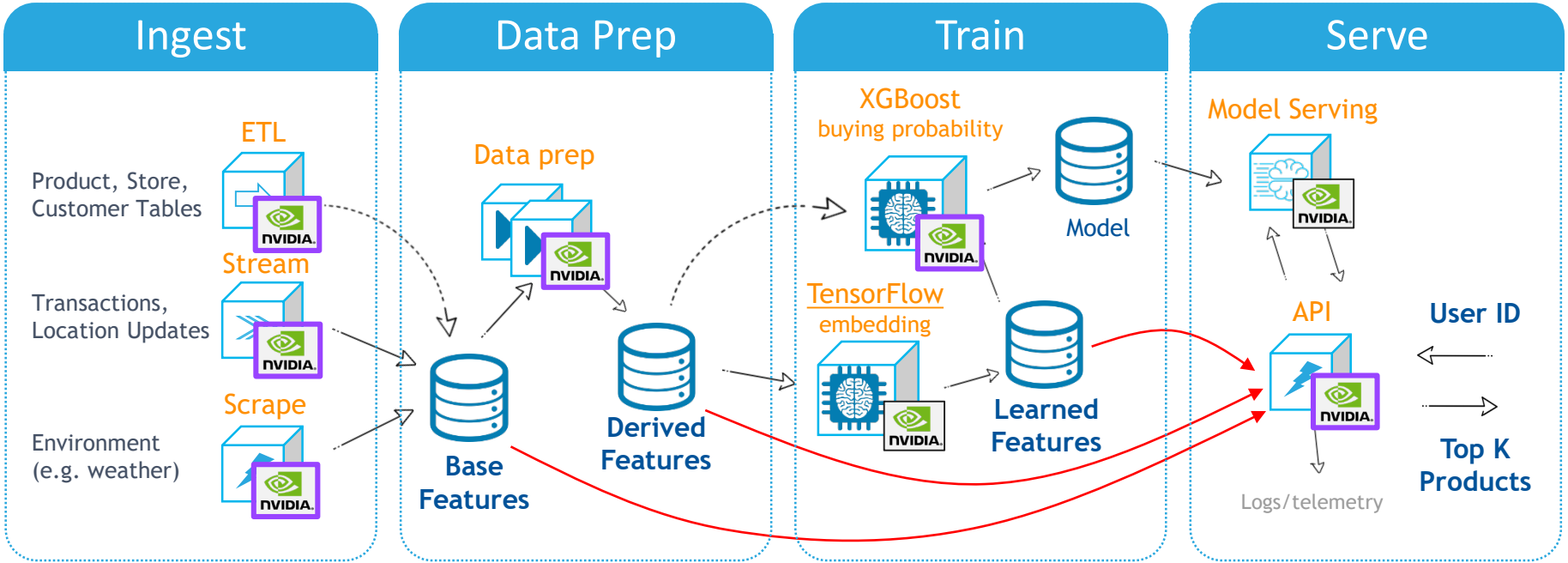
Automation and user facing portals/APIs

Managed Services

Shared GPU/CPU resources

Features (Online, offline) and raw data

Pipeline Example: Real-time Product Recommendations



Elastic functions + GPUs = Performance, Scale, Simplicity

- ML Functions

- Feature/Model Store

RAPIDS

iguazio

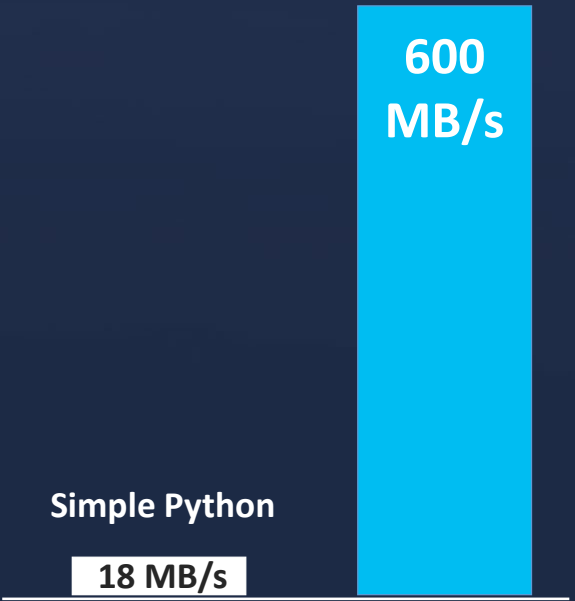
Using Nuclio + GPU to Accelerate ETL and Streaming

https://github.com/nuclio

Simple code! Automated DevOps ! Any Source!
(e.g. read JSON Stream + aggregate + dump to Parquet)

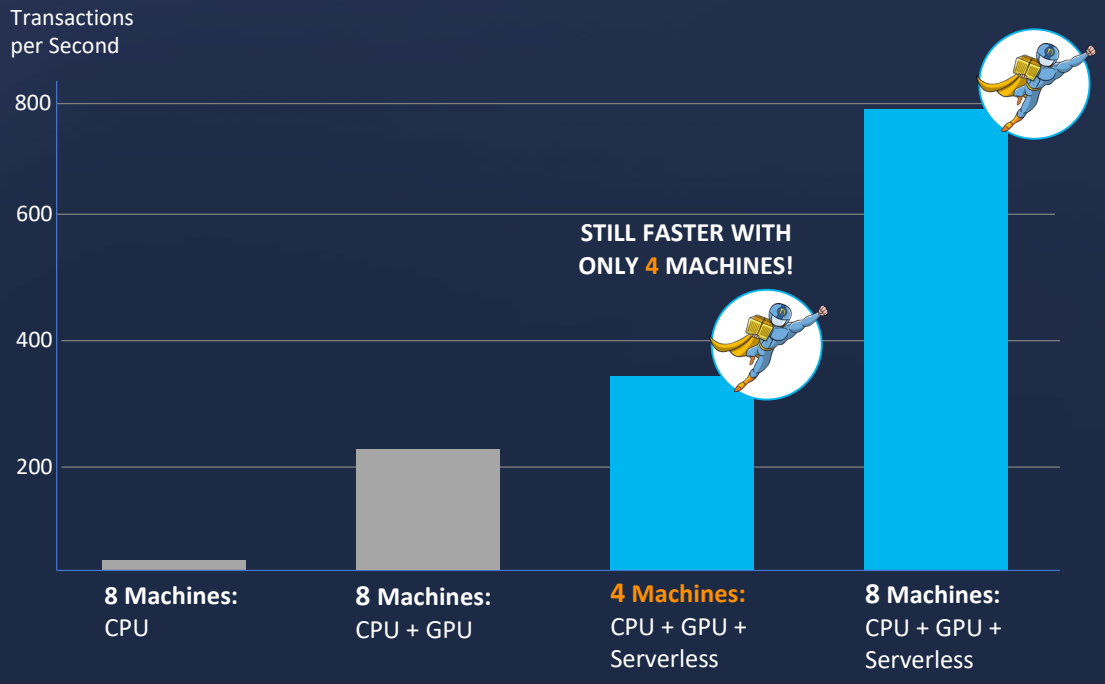
```
def init_context(context):
    os.makedirs(sink, exist_ok=True)

def handler(context, event):
    add_log_to_batch(context, event.body)

    if len(batch) > batch_len:
        df = _batch_to_df(context)
        if not df.empty:
            df = df.groupby(['log_ip']).agg({'feconn':'mean',
                                             'beconn':'mean',
                                             'time_backend_response':'max',
                                             'time_backend_response':'mean',
                                             'time_queue':'mean',
                                             'time_duration': 'mean',
                                             'time_request': 'mean',
                                             'time_backend_connect':'mean'
                                            })
        df_to_parquet(df)
        reset_batch()
```

\+ RAPIDS



iguazio

Using Nuclio for Real-time Model Serving

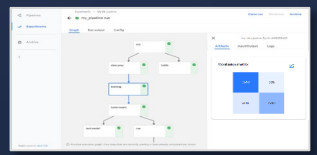
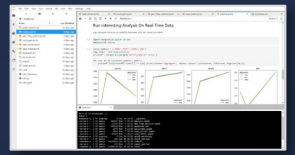
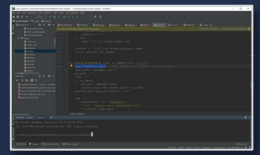
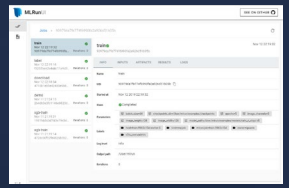


Single command from notebook to function

iguazio

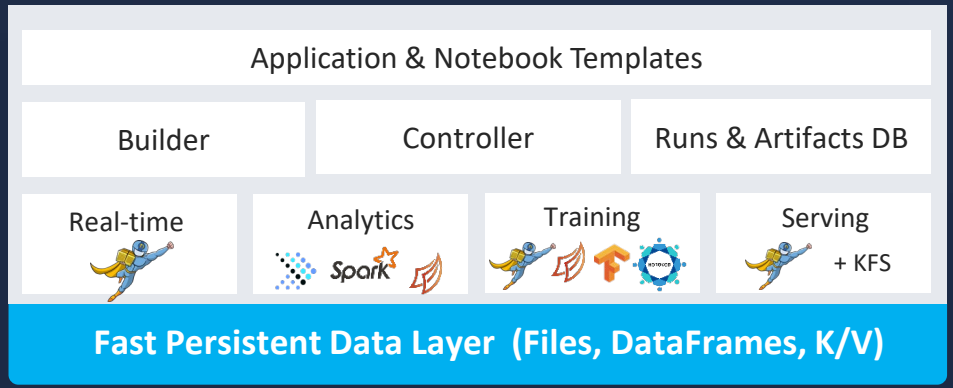
Introducing Nuclio ML Functions and MLRun

Access from your notebook, IDE, or KubeFlow

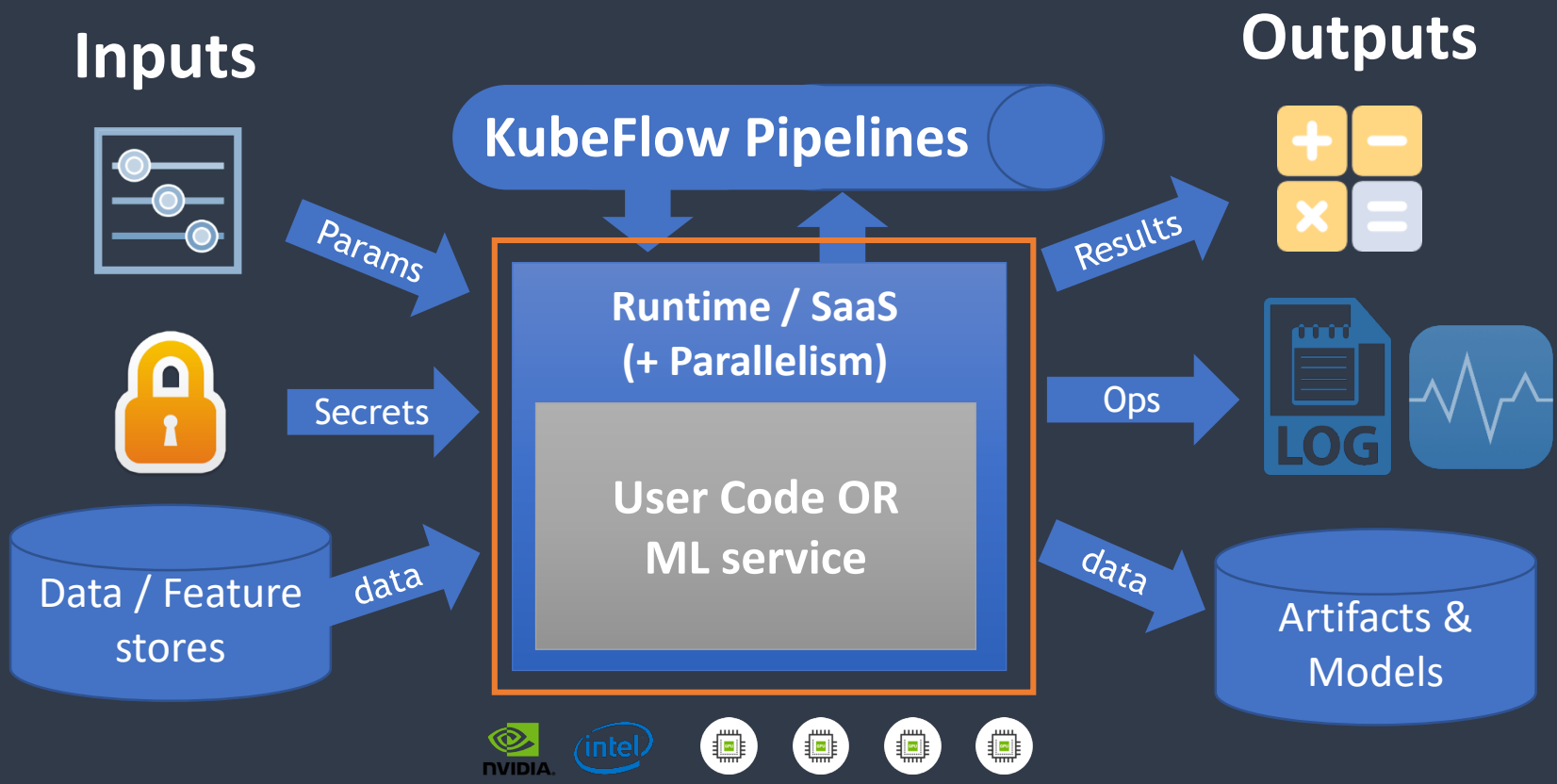
Common APIs & Automation

Multiple Engines



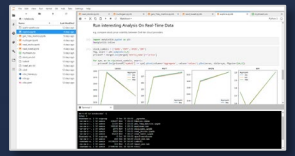
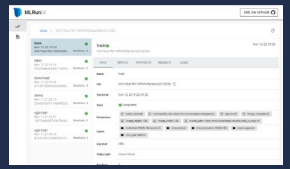
Built-in Artifacts & Runs Tracking

Elastic Scaling of CPUs & GPUs



ML & Analytics Functions Architecture

Demo: Fast and Dynamically Scaling ML Pipeline

Data (Images) Collection › Image Labeling › Distributed Training › Images Classification In Real-time

Real-time Data Fabric

iguazio

Thank You

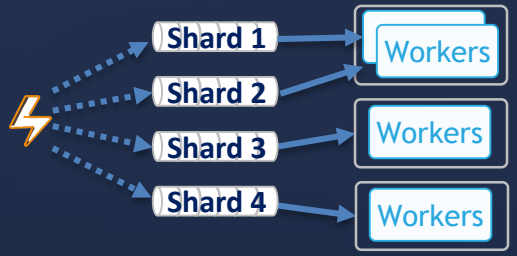
yaronh@iguazio.com, @yaronhaviv

Nuclio: Taking Serverless to Data Intensive Apps

Extreme Performance

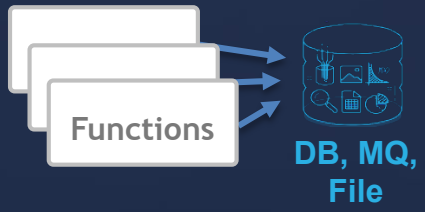
- Non-blocking, parallel
- Zero copy, buffer reuse
- Up to 400K events/sec/proc
- **GPU** optimizations

Advanced Data & AI Features



- Auto-rebalance, checkpoints
- Any source: Kafka, NATS, Kinesis, event-hub, iguazio, pub/sub, RabbitMQ, Cron, ..
- NVIDIA Rapids integration

Statefulness



- Data bindings
- Shared volumes
- Context cache

Kubeflow

Natively integrated with KubeFlow and Jupyter Notebooks

iguazio