

# Building New HTAP Applications: **With Apache Ignite**



Denis Magda  
Apache Ignite PMC Chair  
GridGain Product Management

# In-Memory Computing Best Practices

## Part 2: Build New HTAP Applications

Part 1:

Add Speed and Scale to Existing Applications

Part 2:

Build New HTAP Applications

Part 3:

Automate to Improve Decisions

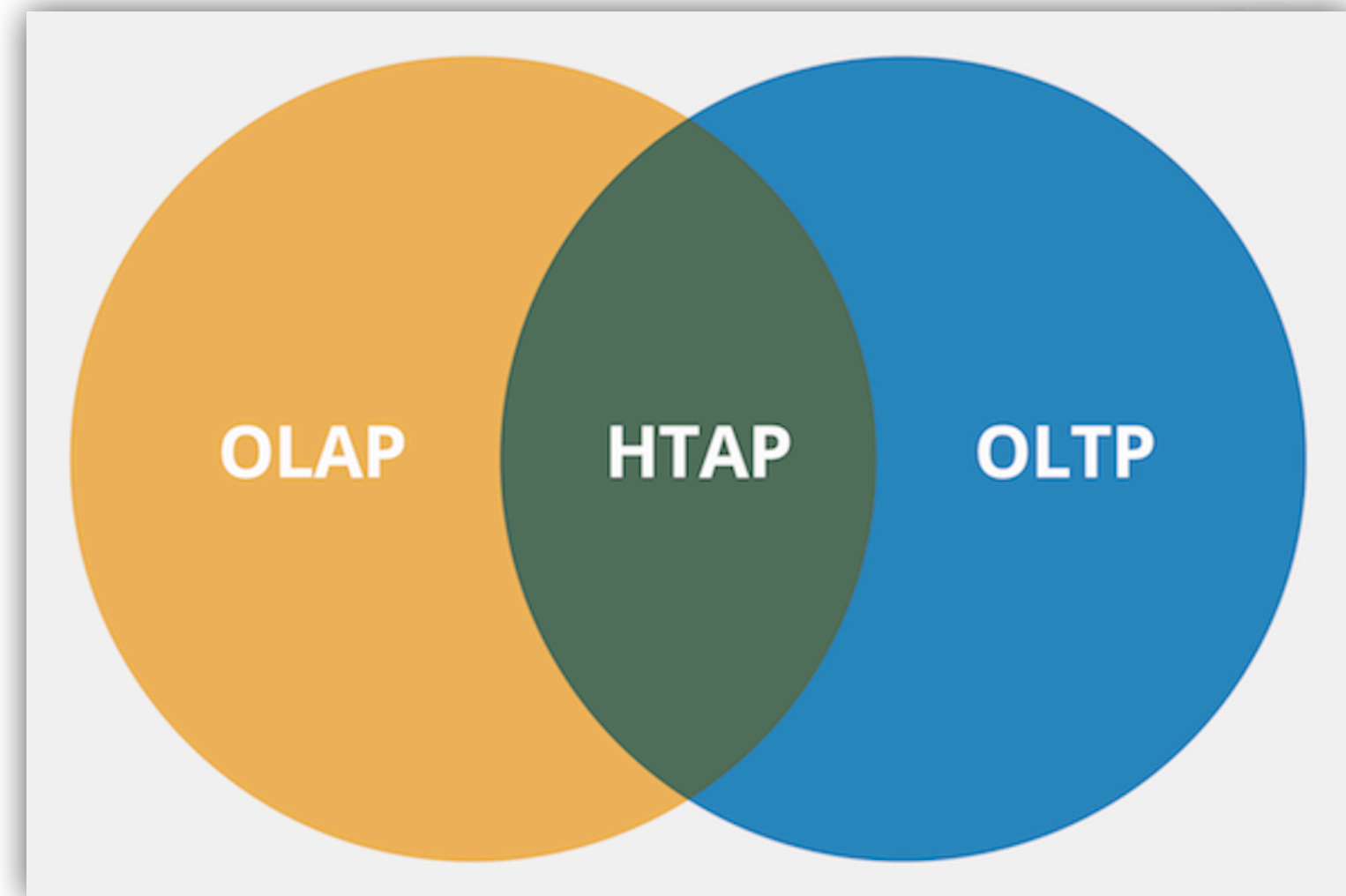
# Agenda

- HTAP Principles
- Apache Ignite for HTAP
  - Memory-Centric Storage
  - Collocated Processing
  - Real-Time Streaming
  - Scalable Machine and Deep Learning
- Demo (Cloud Native Solution)
- Q & A

# Hybrid Transactional/Analytical Processing Principles

# HTAP Principles

- One Platform Multiple Workloads
  - OLTP and OLAP
  - Real-Time Streaming and Batching
  - No ETL
- Distributed Everything
  - Scalability
  - No Single Point of Failure
  - SQL, Calculations, ML, etc.
- Cloud Native



# Apache Ignite for HTAP

# Apache Ignite Overview



Financial Services



Telco



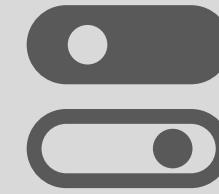
Travel & Logistics



E-Commerce



Pharma & Healthcare



IoT

SQL

Key/Value

Transactions

Compute

Services

Streaming

ML

## Memory-Centric Storage

Scale to 1000s of Nodes & Store TBs of Data

**Ignite Native Persistence**  
(Flash, SSD, Intel 3D XPoint)

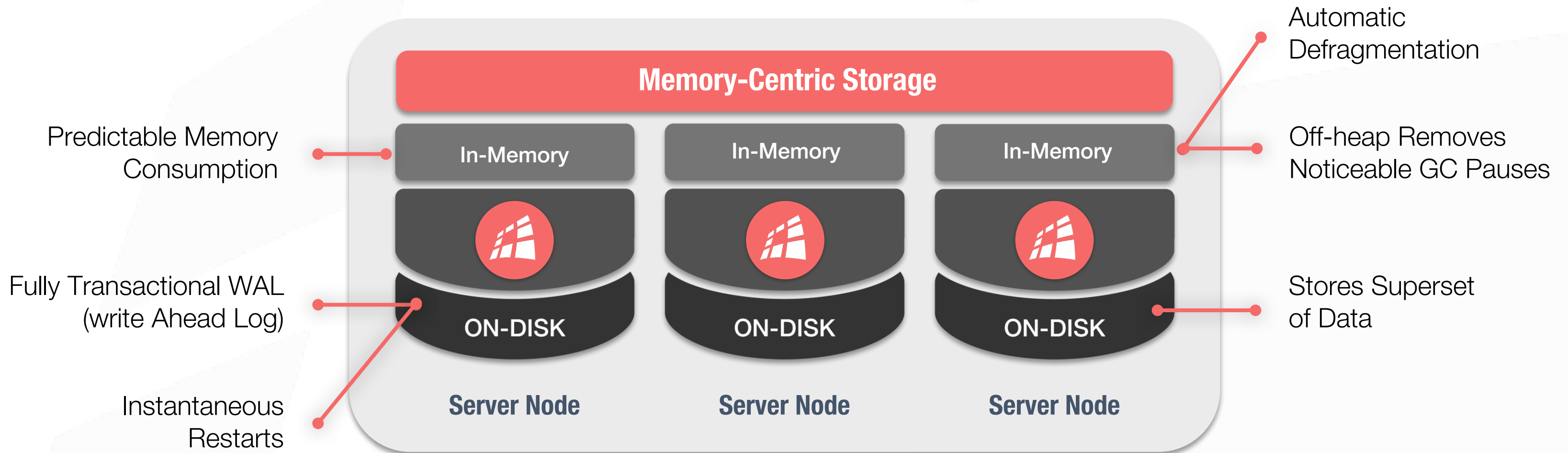
## Third-Party Persistence

Keep Your Own DB  
(RDBMS, HDFS, NoSQL)

# Memory-Centric Storage



# Memory-Centric Storage



# Memory Usage Modes

Mode	Description	Major Advantage
<b>In-Memory</b>	Pure In-Memory Storage	Maximum performance possible (data is never written to disk)
<b>In-Memory + 3<sup>rd</sup> Party DB</b>	Ignite as a caching layer (aka. data grid) above an existing databases – RDBMS, NoSQL.	Reads acceleration and scale of existing solutions
<b>In-Memory + Full Copy on Disk</b>	The whole data set is stored both in memory and on disk	Reads/Writes acceleration and fast restarts
<b>100% on Disk + In-Memory Cache</b>	100% of data is in Ignite native persistence and a subset is in memory	Above + Unlimited scale beyond RAM capacity

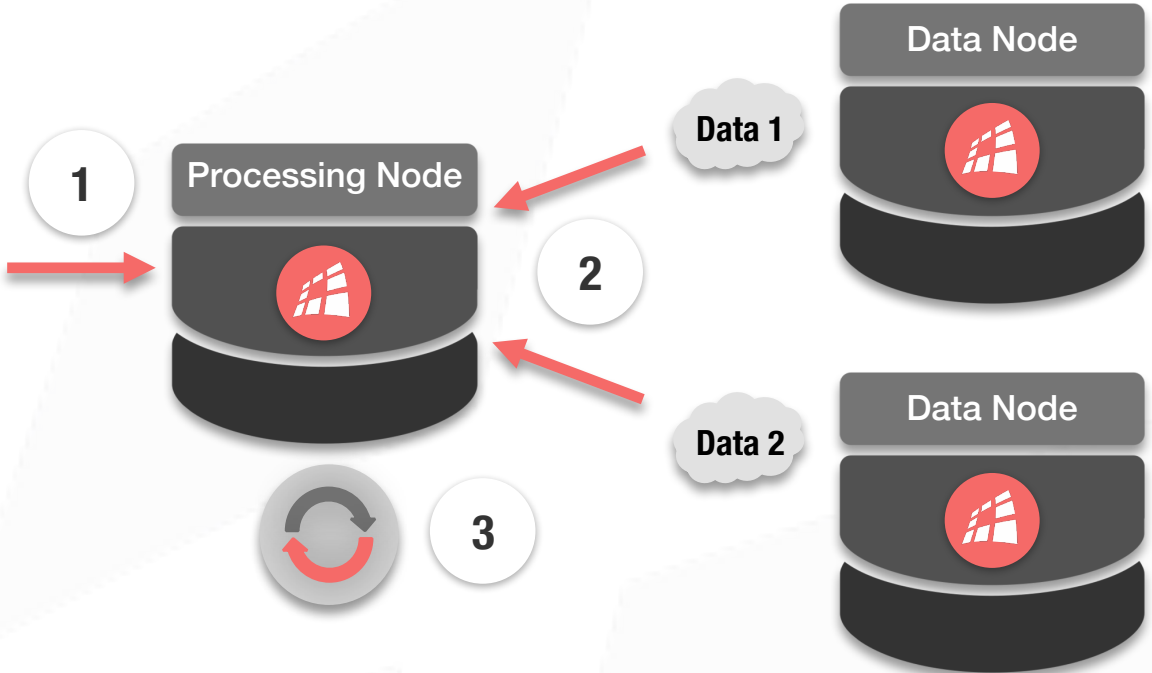
# Collocated Processing

# Affinity Collocation

- Data-to-Data Collocation
  - Countries and Cities, Vendors and Cars
  - Efficient SQL JOINS
- Compute-to-Data Collocation
  - Send Compute to Data (Not Data to Compute)
  - Reduced Network Traffic -> Better Performance

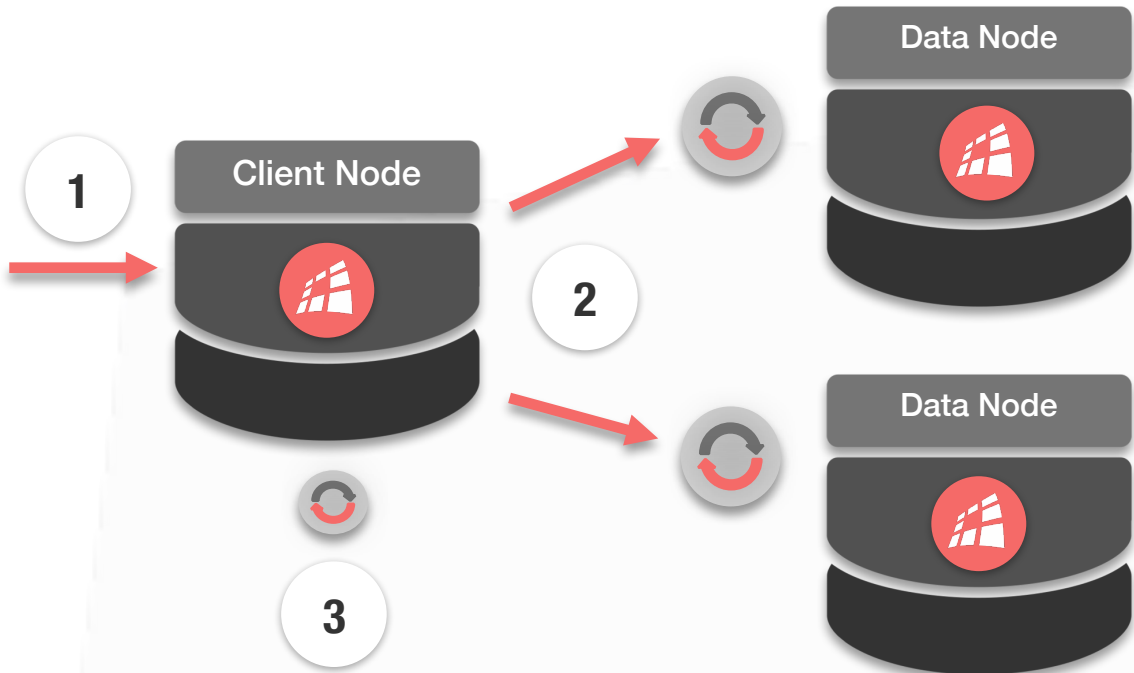


# Client-Server Processing



- 1. Initial Request
- 2. Fetch data from remote nodes
- 3. Process the entire data-set

# Co-located Processing



- 1. Initial request
- 2. Co-locate processing with data
- 3. Reduce multiple results into one

# Collocated Joins



1

```
SELECT ct.name, count(c.name)
FROM Country as ct
JOIN City as c ON c.countryCode = ct.code
WHERE ct.name IN ('Canada', 'India') GROUP BY (ct.name);
```

2

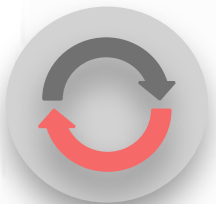


Ignite Node

**Canada**

- Toronto
- Montreal
- Ottawa
- Calgary

2



Ignite Node

**India**

- Mumbai
- New Delhi

3

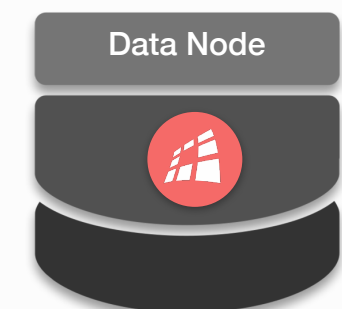
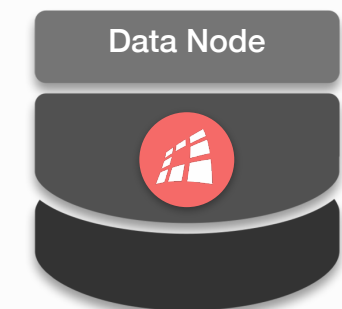
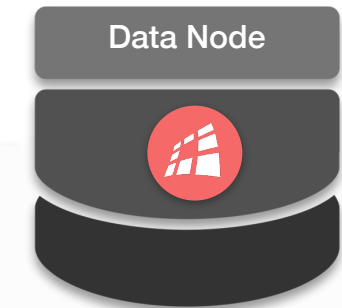


1. Initial Query
2. Query execution over local data
3. Reduce multiple results in one

# Real-Time Streaming

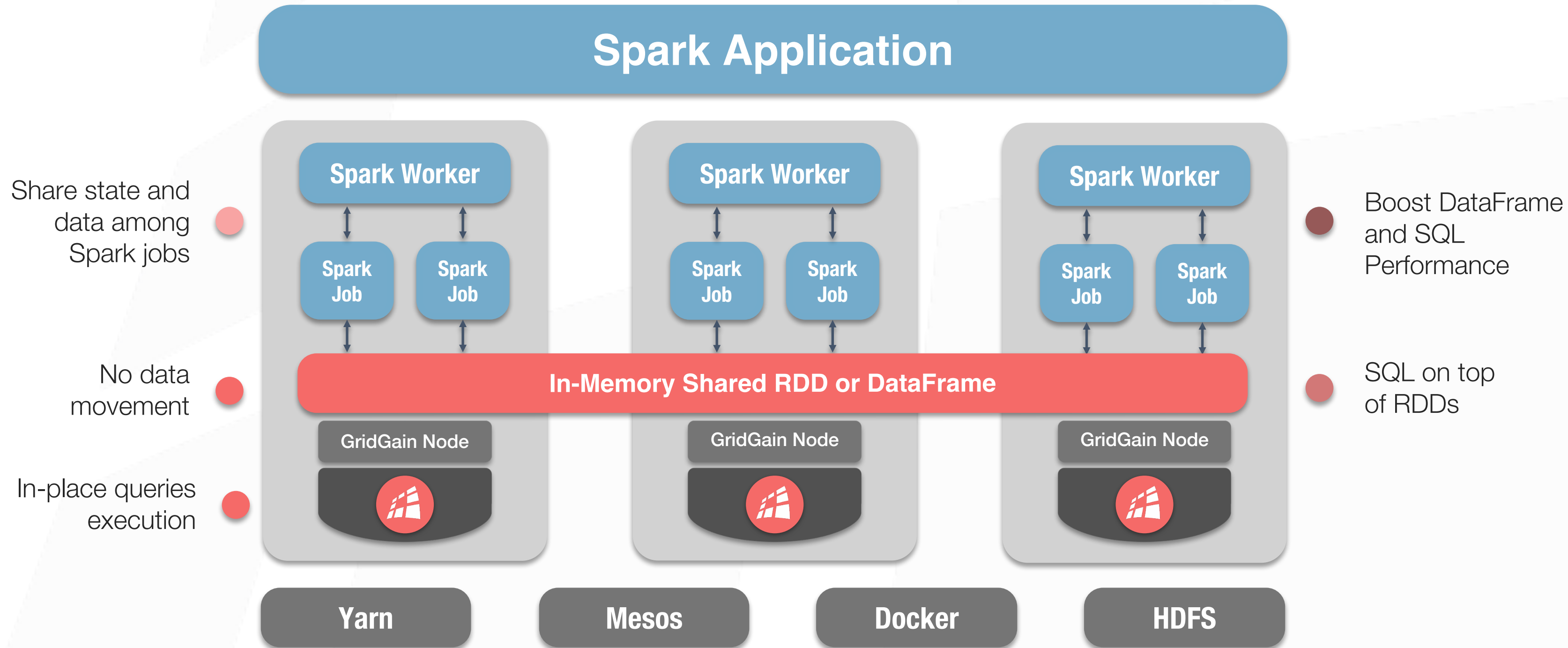
# Real-Time Streaming

- Various Streaming Technologies
  - Kafka, Spark, Flink, Storm, etc.
  - Process, Enrich and then =>
- Ignite as a store for streaming pipelines
  - Streaming Analytics





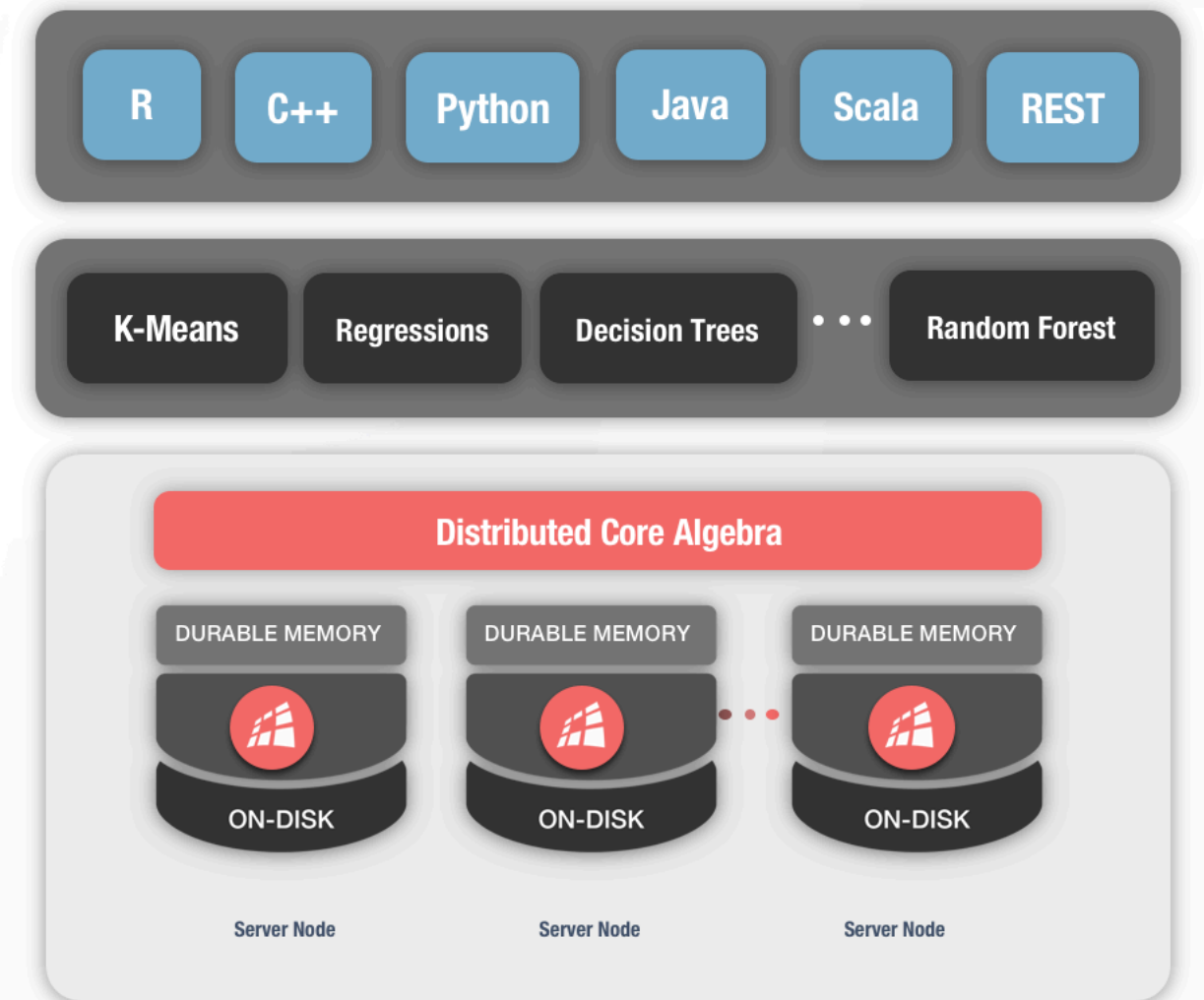
# Ignite + Spark



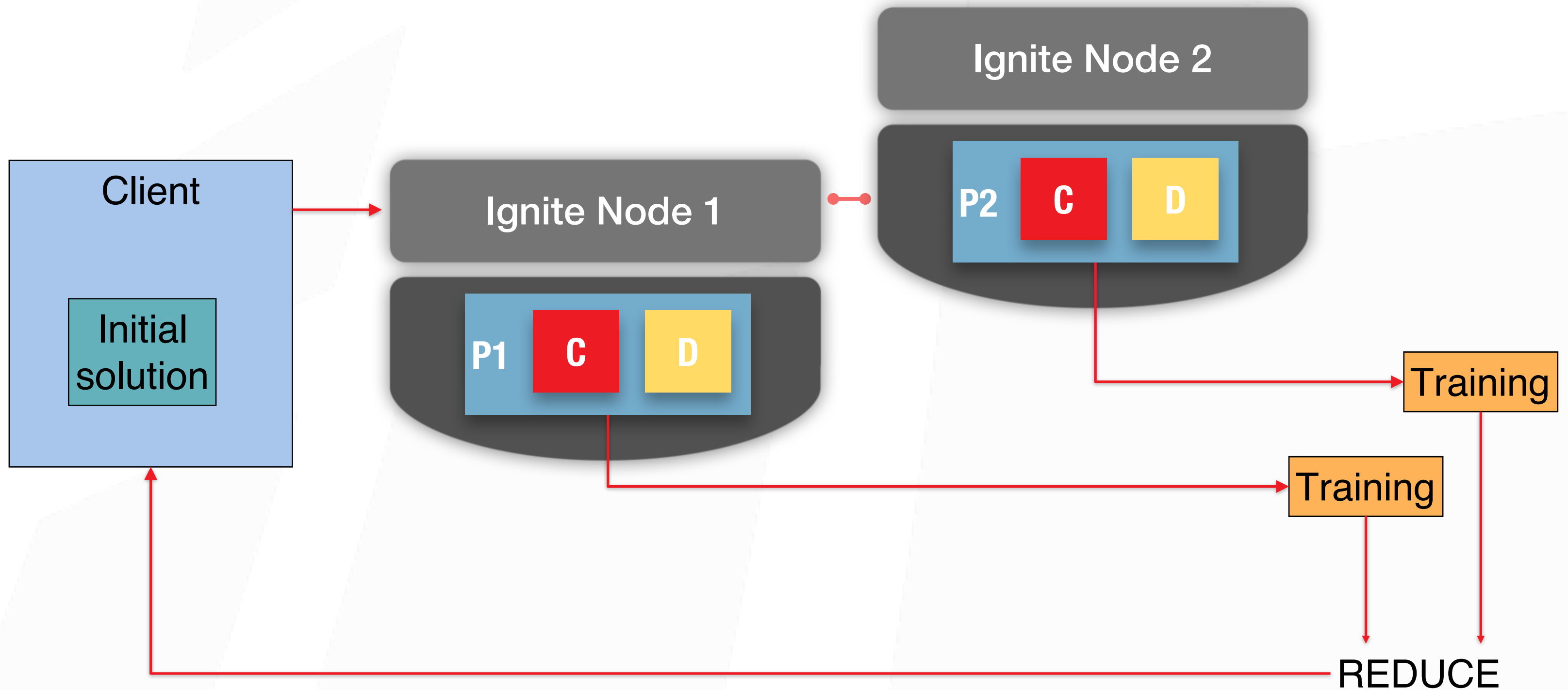
# Scalable ML and DL

# ML/DL for HTAP

- Massive Scalability
  - Scale Beyond Single Server Unit
  - Horizontal + Vertical
  - RAM + Disk
- Zero-ETL
  - Train Models and run Algorithms in Place
- Fault Tolerance and Continuous Learning
  - Partition-based dataset



# Partition-Based Dataset



# Demo - Cloud Native Solutions

The screenshot shows the GridGain web interface. At the top left is the GridGain logo with a 'Beta' badge. The navigation menu includes 'Clusters', 'Queries', 'Monitoring', and 'Documentation'. The user 'Dmitriy Setrakyan' is logged in, and the status 'Connected clusters: 1' is shown in the top right. The main content area is titled 'Clusters' and features a '+ Launch cluster' button. Below this is a 'My Clusters' section with a settings gear icon and a table of clusters. The table has columns for Name, Plan, Size, Uptime, Status, and Control. One cluster, 'Dmitriy's Demo Cluster', is listed with plan 'aws/us-west-1/t2.micro', size '5', uptime '1d 4h 26m', and status 'Running'. Below the table is a 'Demos' section with tabs for 'Quick Demo' and 'Command Line'. The 'Command Line' tab is active, showing a progress bar with five steps: 1. Download, 2. SSL, 3. Connect, 4. Preload, and 5. Query. Underneath is a 'Run SQL Queries' section with a text area containing the SQL query: 'SELECT name, population FROM city ORDER BY population DESC LIMIT 3;'. A link 'here' is provided for running sample queries.

**GridGain** Beta

Clusters Queries Monitoring Documentation Connected clusters: 1 Dmitriy Setrakyan

## Clusters

+ Launch cluster

My Clusters

Name	Plan	Size...	Uptime	Status	Control
Dmitriy's Demo Cluster	aws/us-west-1/t2.micro	5	1d 4h 26m	Running	

Demos

Quick Demo **Command Line**

1. Download > 2. SSL > 3. Connect > 4. Preload > 5. Query

Run SQL Queries

- Execute SQL query:

```
SELECT name, population FROM city
ORDER BY population DESC LIMIT 3;
```
- To run sample queries click [here](#)

<https://cloud.gridgain.com>

# Summary: HTAP and Apps Architecture

- Single storage and platform
  - OLAP, OLTP, Streaming
- No ETL
- Horizontal Scalability is at the Core
- Cloud Deployment Options



# Thank You!!!

Thank you for joining us. Follow the conversation.

<https://ignite.apache.org>



@denismagda  
#apacheignite  
#gridgain