How to Add Speed and Scale to SQL, Support New Data Needs, and Keep Your RDBMS

Valentin Kulichenko
Apache Ignite PMC Member
GridGain Lead Architect

GridGain Platform Overview

GridGain In-Memory Computing Platform

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)

Data Snapshots & Recovery

Monitoring & Management

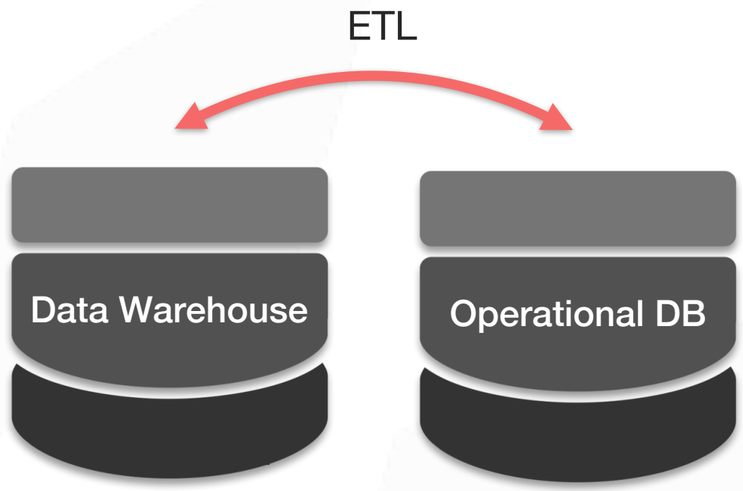
Security & Auditing

Datacenter Replication

GridGain

GridGain HTAP Architecture

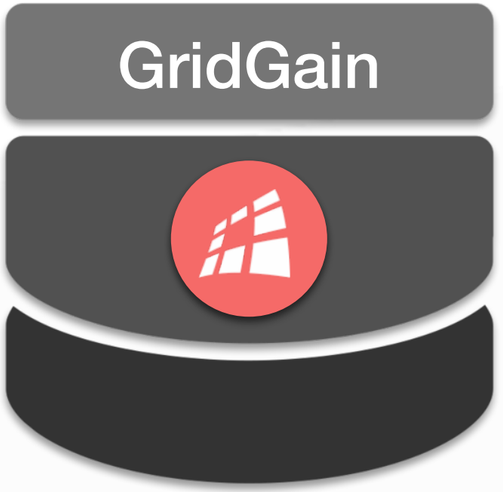
"IMC-enabled HTAP can have a transformational impact on the business." — Gartner 2/17



Analytics, ML, AI

Transactions

ETL, Batch, Inflexible



Analytics, ML, AI & Transactions

Real Time, Scalable,
Available, Flexible

GridGain Systems Customers

Financial Services

SBERBANK
citibank
ING
BARCLAYS
APOLLO
Jefferies
Fidelity INVESTMENTS
AMERICAN EXPRESS

FinTech

MISYS
IHS Markit
THE GLUE FINTECH SOLUTIONS
SS&C ADVENT

Software

SS&C ADVENT
Microsoft
workday.
Siemens PLM Software
SIEMENS
IBM

Telco

HUAWEI
RingCentral
CYBER DUST
intelligentpipe

Pharma & Healthcare

PRIME THERAPEUTICS
e-Therapeutics plc

Logistics & Travel

Nederlandse Spoorwegen
MercuryGate TMS that delivers.
American Airlines
pitney bowes

E-commerce

24 FITNESS
newegg
JacTravel

IoT

thingworx
Silver Spring NETWORKS

Adtech

adalyser
nielsen

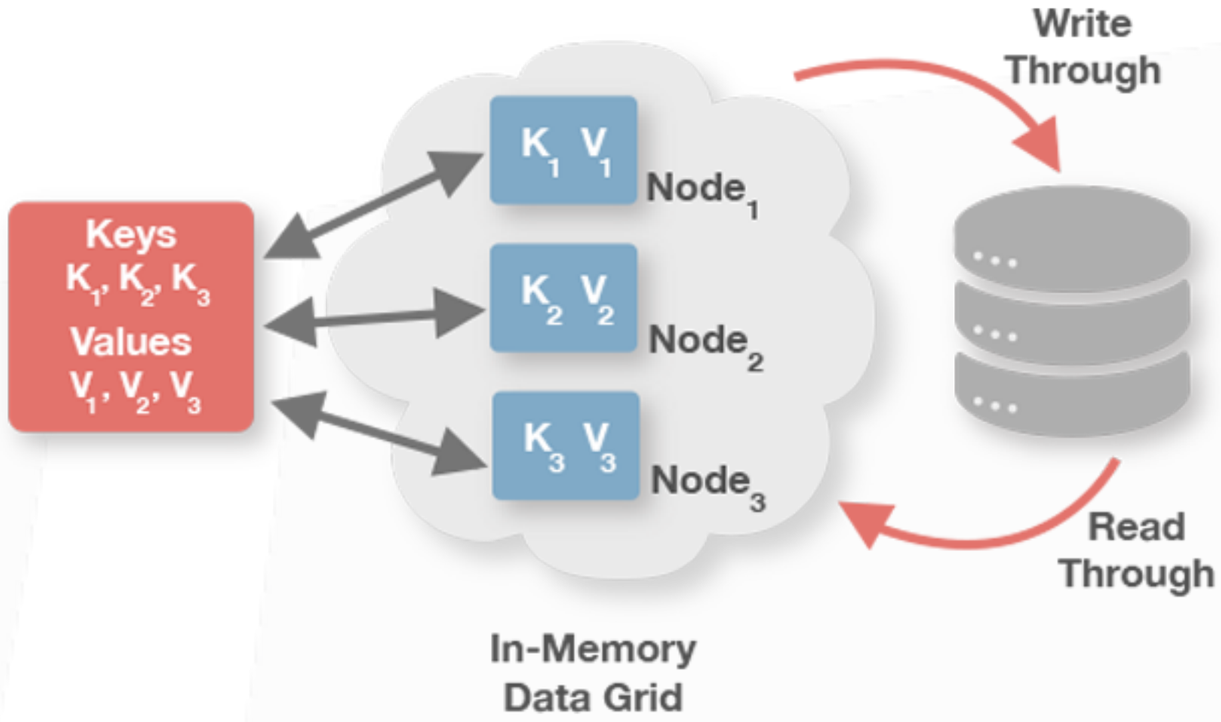
Accelerating SQL

Memory & Disk Utilization

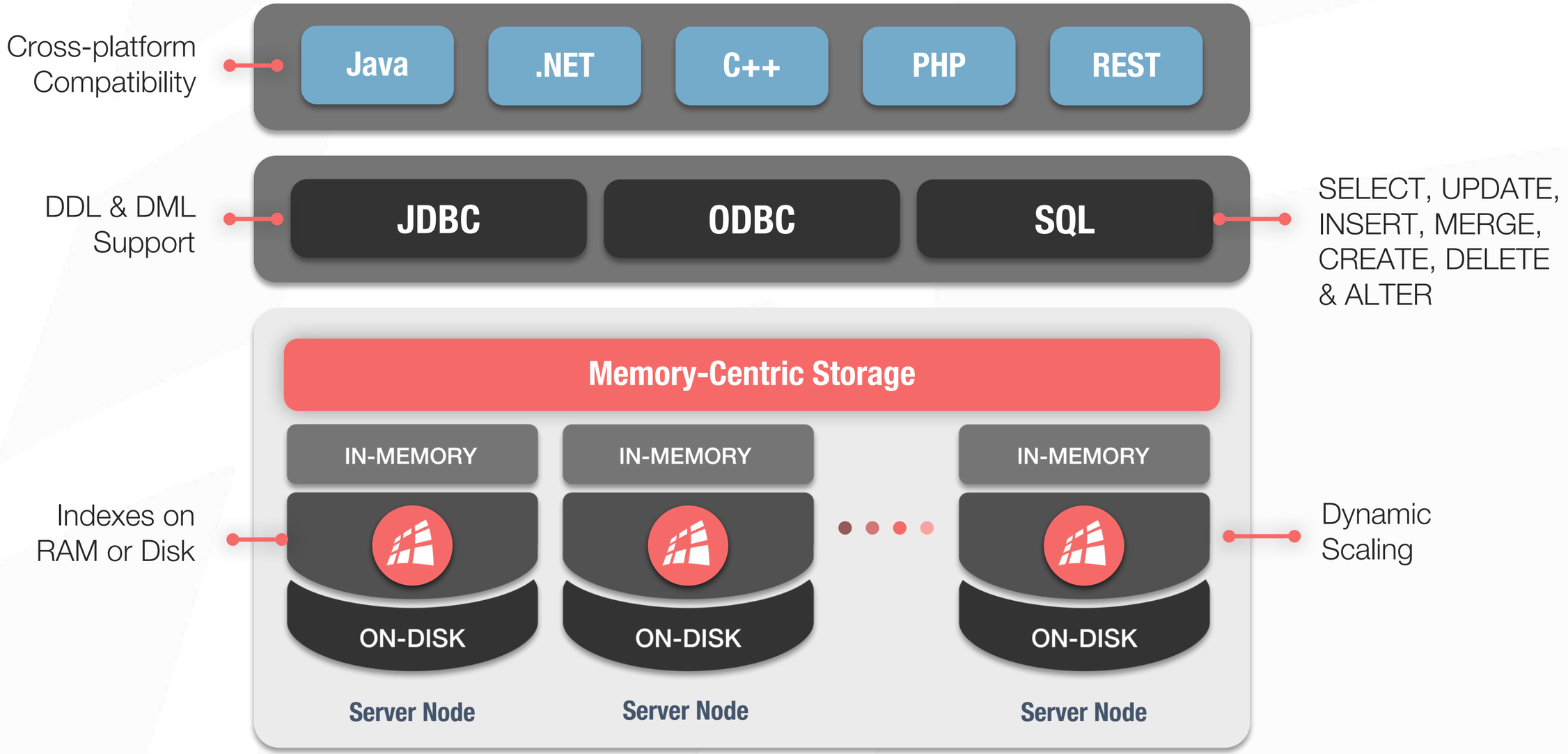
Mode	Description	Major Advantage
In-Memory	Pure In-Memory Storage	Maximum perfomance possible (data is never written to disk)
In-Memory + 3rd Party DB	Caching layer (aka. in-memory data grid) for existing databases – RDBMS, NoSQL, etc	Horizontal scalability Faster reads and writes
In-Memory + Full Copy on Disk	The whole data set is stored both in memory and on disk	Survives cluster failures
100% on Disk + In-Memory Cache	100% of data is in Ignite native persistence and a subset is in memory	Unlimited data scale beyond RAM capacity

Turbocharging Database System

- Database Caching Use Case
  - Slide Ignite in between Database System and applications
- No ‘rip and replace’ Performance Boost
  - Keep data both in memory and Database System
  - Scale to 1000s of nodes
- Automatic Read-Through and Write-Through
  - Key-Value Operations Only
- ANSI-99 SQL
  - Over in-memory data sets



Distributed SQL



Connectivity

- JDBC
- ODBC
- REST
- Java, .NET and C++ APIs

```
// Register JDBC driver.
Class.forName("org.apache.ignite.IgniteJdbcThinDriver");

// Open the JDBC connection.
Connection conn = DriverManager.getConnection("jdbc:ignite:thin://192.168.0.50");
```

```
./sqlline.sh --color=true --verbose=true -u jdbc:ignite:thin://127.0.0.1/
```

Data Definition Language

- CREATE/DROP TABLE
- CREATE/DROP INDEX
- ALTER TABLE
- Changes Durability
  - Ignite Native Persistence

```
CREATE TABLE `city` (
  `ID` INT(11),
  `Name` CHAR(35),
  `CountryCode` CHAR(3),
  `District` CHAR(20),
  `Population` INT(11),
  PRIMARY KEY (`ID`, `CountryCode`)
) WITH "template=partitioned, backups=1, affinityKey=CountryCode";
```

https://apacheignite-sql.readme.io/docs/ddl

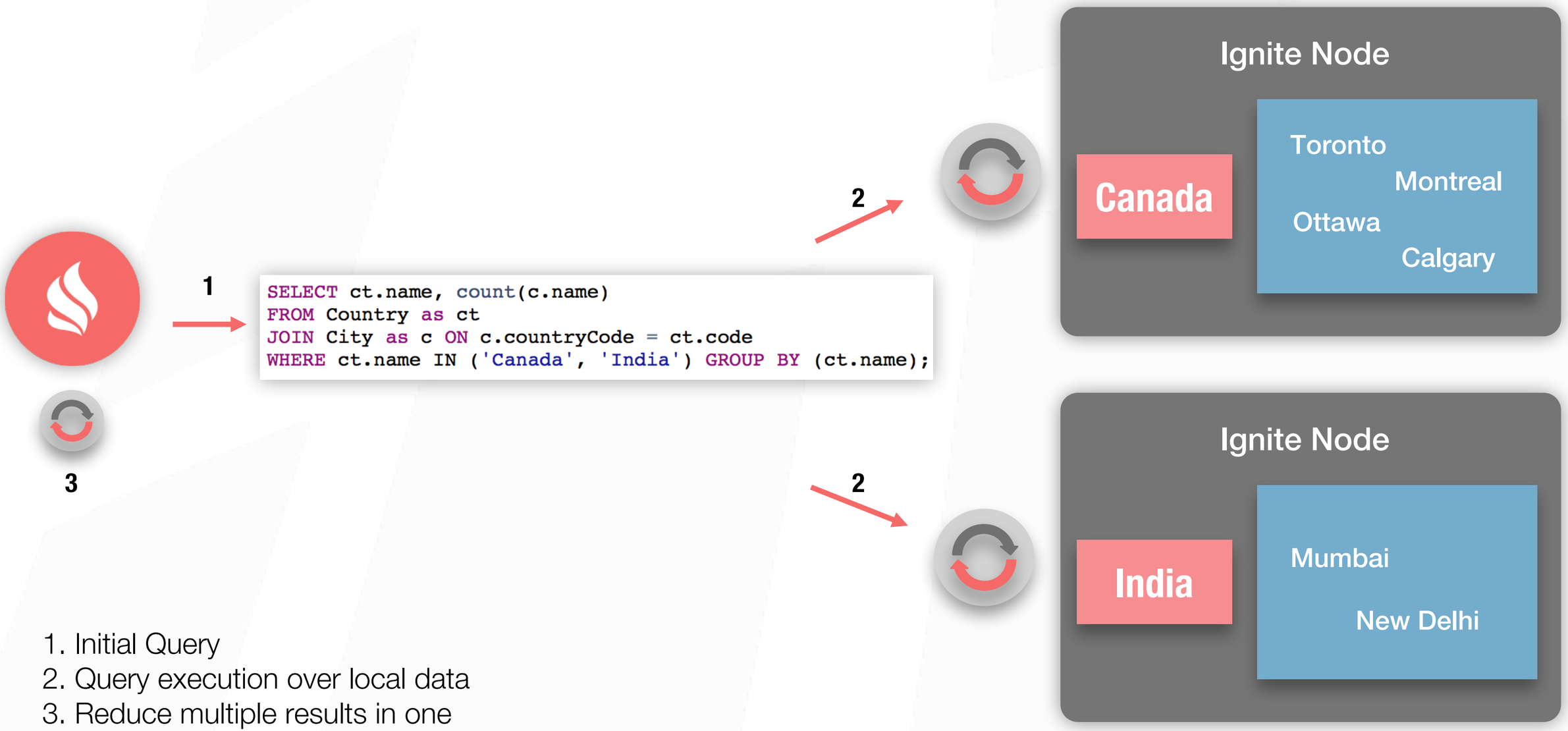
Data Manipulation Language

- ANSI-99 specification
- Fault-tolerant and consistent
- INSERT, UPDATE, DELETE
- SELECT
  - JOINs
  - Subqueries

```
SELECT country.name, city.name, MAX(city.population) as max_pop
FROM country JOIN city ON city.countrycode = country.code
WHERE country.code IN ('USA','RUS','CHN')
GROUP BY country.name, city.name ORDER BY max_pop DESC LIMIT 3;
```

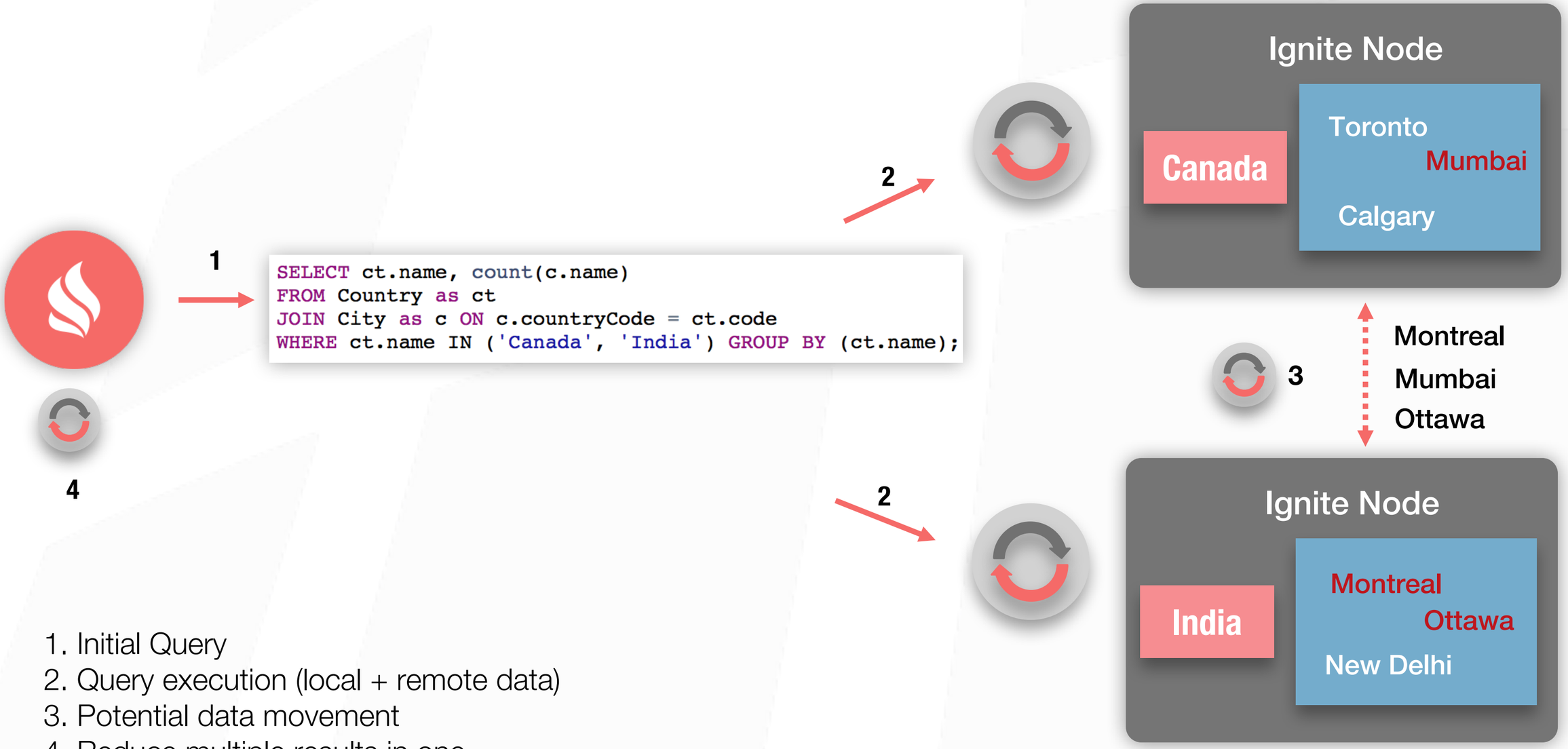
https://apacheignite-sql.readme.io/docs/dml

Collocated Joins



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

Non-Collocated Joins



1. Initial Query
2. Query execution (local + remote data)
3. Potential data movement
4. Reduce multiple results in one

Demo

Ignite as Memory-Centric Database

Memory & Disk Utilization

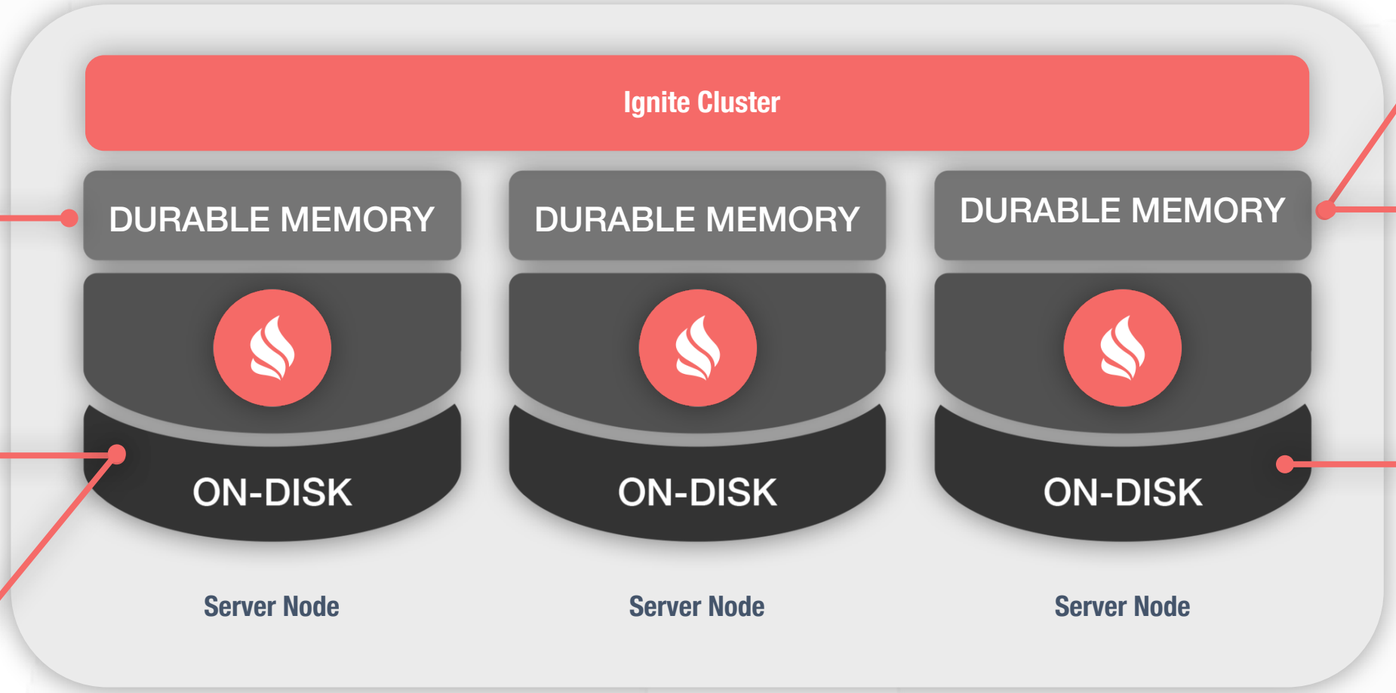
Mode	Description	Major Advantage
In-Memory	Pure In-Memory Storage	Maximum perfomance possible (data is never written to disk)
In-Memory + 3rd Party DB	Caching layer (aka. in-memory data grid) for existing databases – RDBMS, NoSQL, etc	Horizontal scalability Faster reads and writes
In-Memory + Full Copy on Disk	The whole data set is stored both in memory and on disk	Survives cluster failures
100% on Disk + In-Memory Cache	100% of data is in Ignite native persistence and a subset is in memory	Unlimited data scale beyond RAM capacity

Durable Memory

Predictable memory consumption

Fully Transactional (Write-Ahead Log)

Instantaneous Restarts



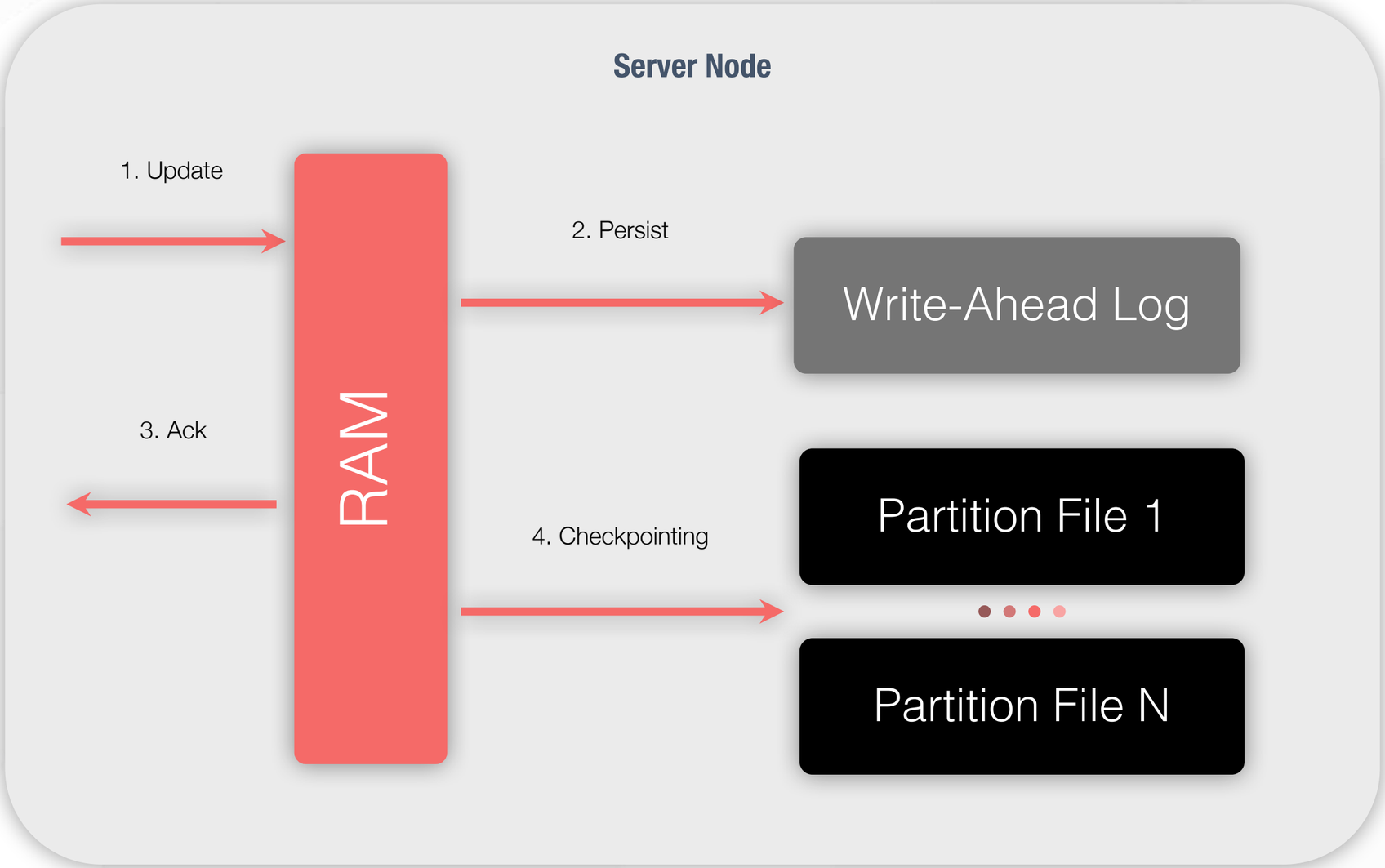
Automatic Defragmentation

Off-heap Removes noticeable GC pauses

Stores Superset of Data

GridGain

Ignite Native Persistence



Any Questions?

Follow the conversation.
http://www.gridgain.com

#apacheignite
#gridgain
@vkulichenko