GridGain

Getting Started with Apache Ignite as an In-Memory Database (IMDB)

Glenn Wiebe

Senior Solution Architect

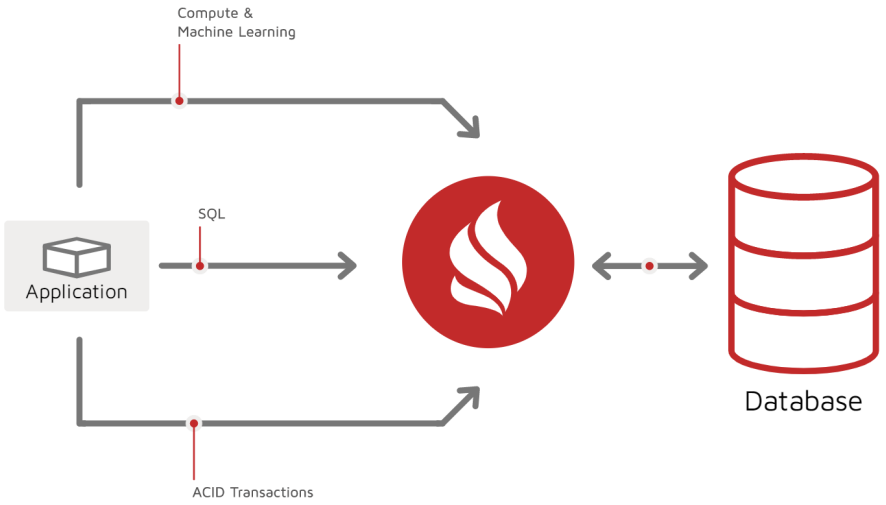
July 8, 2020

Getting Started with Ignite IMDB - Agenda

- **Patterns {Features}** (5 minutes)
- **Data Load Facilities** (4 minutes)
- **IMDB Project Process** (1 minutes)
- **IMDB Project Build Demo** (30 minutes)
- **Q&A** (5 minutes)

Getting Started with Ignite IMDB – Ignite Patterns

1. **Ignite Cache** – JCache or JSR 107 basic caching API plus added features:
   - – Caches
   - – Cache managers
   - – Cache providers, and
   - – Entries (i.e. Key-Value pairs)
   - – Entry Processors
   - \+ Transactions (atomic & fully transactional)
   - \+ Multiple fine-grained API extensions, e.g.:
     `getAndPutIfAbsentAsync(K key, V val)`
   - \+ Full SQL-99 DML & DDL
   - \+ Scan, SQL & Text Queries
   - \+ Cache Stores: Plug-in 3rd Party & Native Persistence

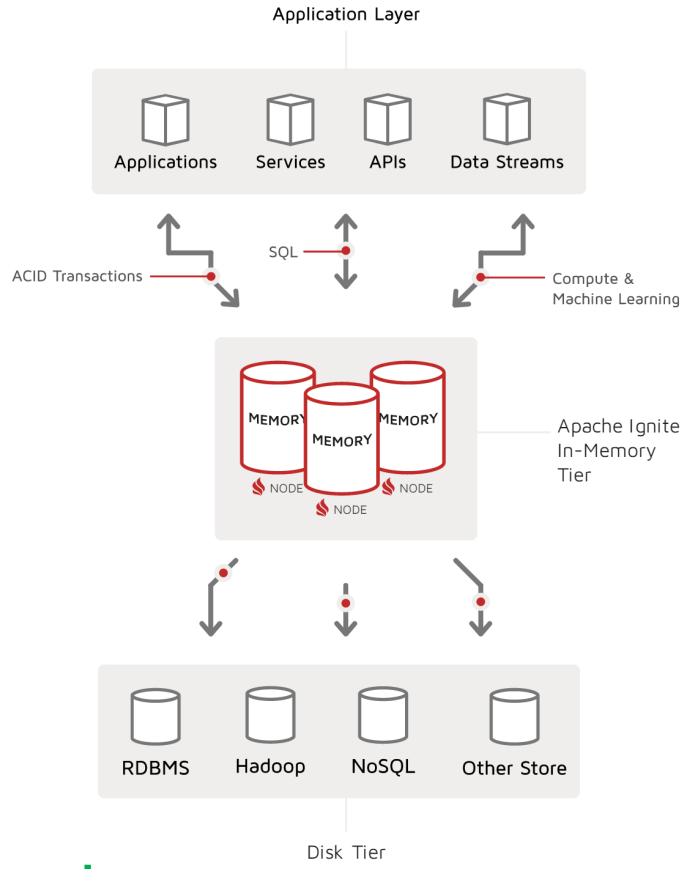


← Ignite is bi-modal: Caches with key-value API can also be Tables with SQL

Getting Started with Ignite IMDB – Ignite Patterns

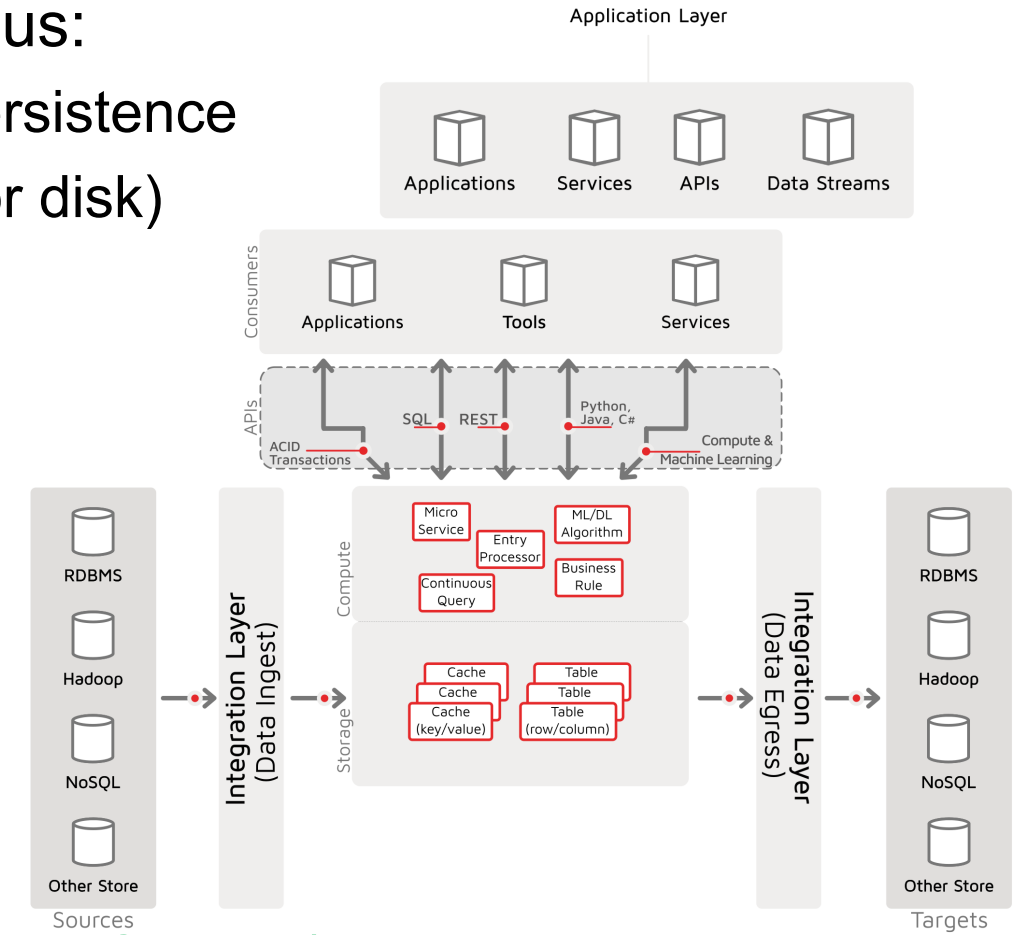
2. **IMDG** – In Memory Data Grid, a ***distributed*** pattern for deploying caches with:
   - Distributed multi-node cluster, with
   - Strongly consistent data platform (see CAP), with
   - Client API access (SQL & Key-Value)
   - Multi-technology integration (Java, C#, .NET, Python, etc.)
   - Advanced API extensions including:
   + Session & Session Management
   + Messaging (pub/sub, ordered & directed)
   + Advanced query & compute processing models, including:
   + Co-located compute, Map-Reduce/Split-Join semantics
   + Automated Read-through & Write-through

   ← For IMDG caches/tables are SYNCHRONIZED with Data Store!



Getting Started with Ignite IMDB – Ignite Patterns

3. **IMDB** – Ignite In Memory Database, IMDG plus:
   - Durable Memory Storage via Ignite Native Persistence
   - Multiple storage tiers (named regions: RAM or disk)
   - Instantaneous Re-start (no cache warmup)
4. **Digital Integration Hub** – IMDB plus:
   - Integration Facilities, including
   - Data Streamer
   - Tools (e.g. SQL Line)
   - Continuous Queries & Cache Interceptors
   - Event-based Processing, including
   - Services, ML/DL, Rules Engine (CEP)



← For IMDB & DIH Data Sources INTEGRATE with Caches/Tables

Getting Started with Ignite IMDB – Data Load Facilities

1. **Ignite APIs** – Cache put(k,v), putAll(map<k,v>) and SQL INSERT
   - Fine-grained APIs with sync and async semantics
   - Transactional and Ordered processing (for cache APIs, SQL MVCC is beta)

   ← Useful in application interfaces with incremental real-time integration patterns
2. **Ignite DataStreamer** – addData(k,v) or addData(map<k,v>)
   - Buffered, Multi-threaded writes
   - Topology aware dispatching to minimize network IO and memory impact
   - Extensible receiver logic
   - At-Least Once loading guarantees
   - Does not support transaction semantics, but
     Implicit and user-defined explicit transactionality can be applied (no order guarantee)

   ← Fastest load mechanism supporting both bulk and incremental feeds

Getting Started with Ignite IMDB – Data Load Facilities

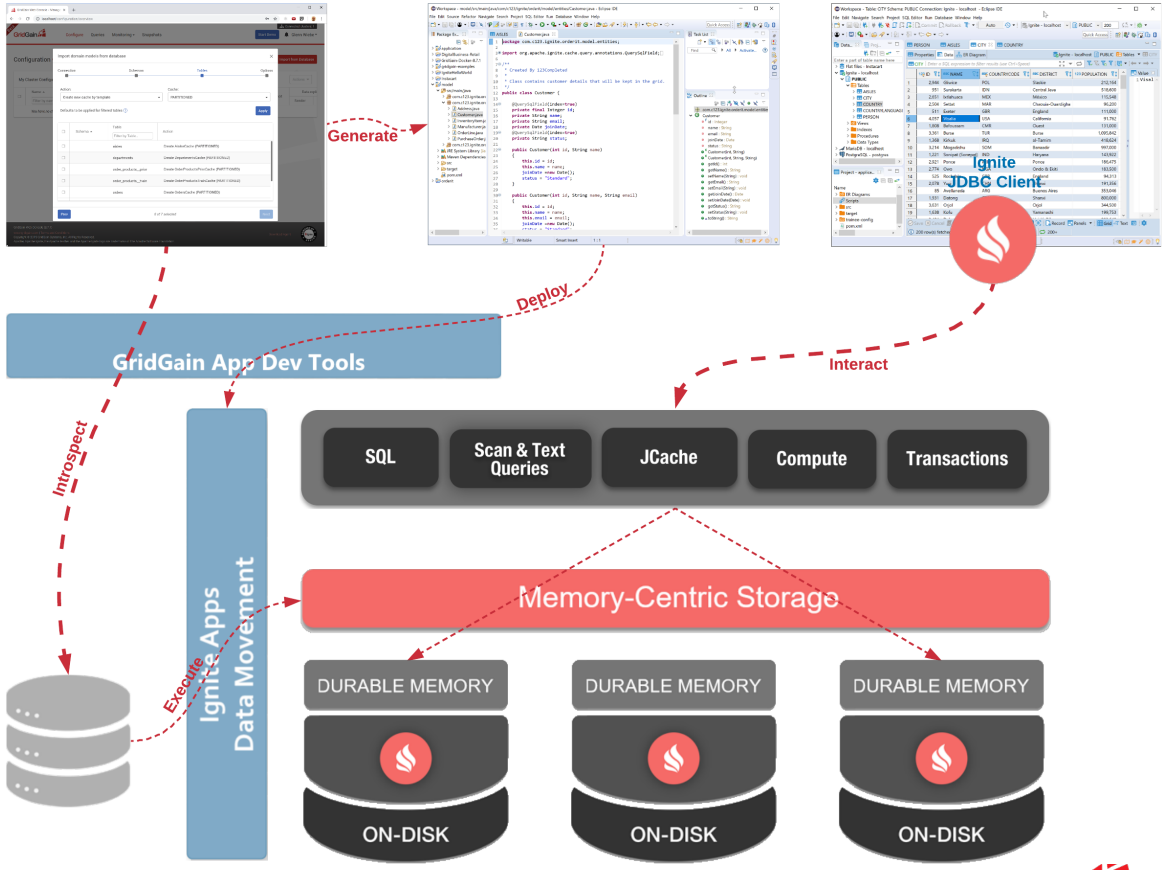
3. **IgniteCache.loadCache()** – cache store implementation to load from source
   - Executed on cache's configured CacheStore on each local node
   - Distributed, node-local loading pattern (i.e. client only initiates load, does no loading)
   - Supports IMDB integration (vs IMDG synchronization) via LoadOnly cache store type

   ← Useful for bulk, initial or batch loads, also event-driven loads (quasi incremental)
4. **Ignite & 3rd Party Tools** – sqlline, DBeaver, Talend, Informatica, etc.
   - Usually SQL JDBC/ODBC based
   - Supports streaming
   - With latest JDBC thin client, supports partition awareness (for directed writes and no intranode data shuffle)

   ← Great for integrating with existing tooling and organizational processes (and no code)

Getting Started with Ignite IMDB – Project Process

1. **Introspect** existing assets using GridGain Web Console Import Wizard
2. **Generate** Maven project using GridGain Web Console
3. **Customize, Build, Deploy & Run** cluster
4. **Execute** Data Load (i.e. ETL or data ingest)
5. **Interact** with IMDB (e.g. jdbc /odbc SQL client, python, spark, etc.)



Getting Started with Ignite IMDB – Demo Part 1

Getting Started with Ignite IMDB – Demo Part 1

1. **Configure** Ignite cluster using GridGain Web Console
2. **Model** Data using GridGain Web Console & Import DB Wizard
3. **Examine** Maven Ignite IMDB cluster project download

🡐 Great jump start or template for building Ignite solutions that scale!

Getting Started with Ignite IMDB – Demo Part 2

Getting Started with Ignite IMDB – Demo Part 2

1. **Configure** Ignite cluster using GridGain Web Console
2. **Model** Data using GridGain Web Console & Import DB Wizard
3. **Examine** Maven Ignite IMDB cluster project download
4. **Customize** Ignite IMDB cluster project
5. **Extend** Ignite IMDB cluster project with data loading components
6. **Build** Ignite IMDB cluster project (Maven build lifecycle)
7. **Deploy** Ignite IMDB cluster

← You are a developer, the options here are virtually unlimited…

Make your {dev-run-test} loop fast and tight!

Getting Started with Ignite IMDB – Demo Part 3

Getting Started with Ignite IMDB – Demo Part 3

1. **Configure** Ignite cluster using GridGain Web Console
2. **Model** Data using GridGain Web Console & Import DB Wizard
3. **Examine** Maven Ignite IMDB cluster project download
4. **Customize** Ignite IMDB cluster project
5. **Extend** Ignite IMDB cluster project with data loading components
6. **Build** Ignite IMDB cluster project (Maven build lifecycle)
7. **Deploy** Ignite IMDB cluster
8. **Run & Test** Ignite IMDB cluster

Getting Started with Ignite IMDB – Q&A