Apache Ignite™ 2.0

Prelude to a Distributed SQL Database

Denis Magda
GridGain Product Manager
Apache Ignite PMC

http://ignite.apache.org

#apacheignite

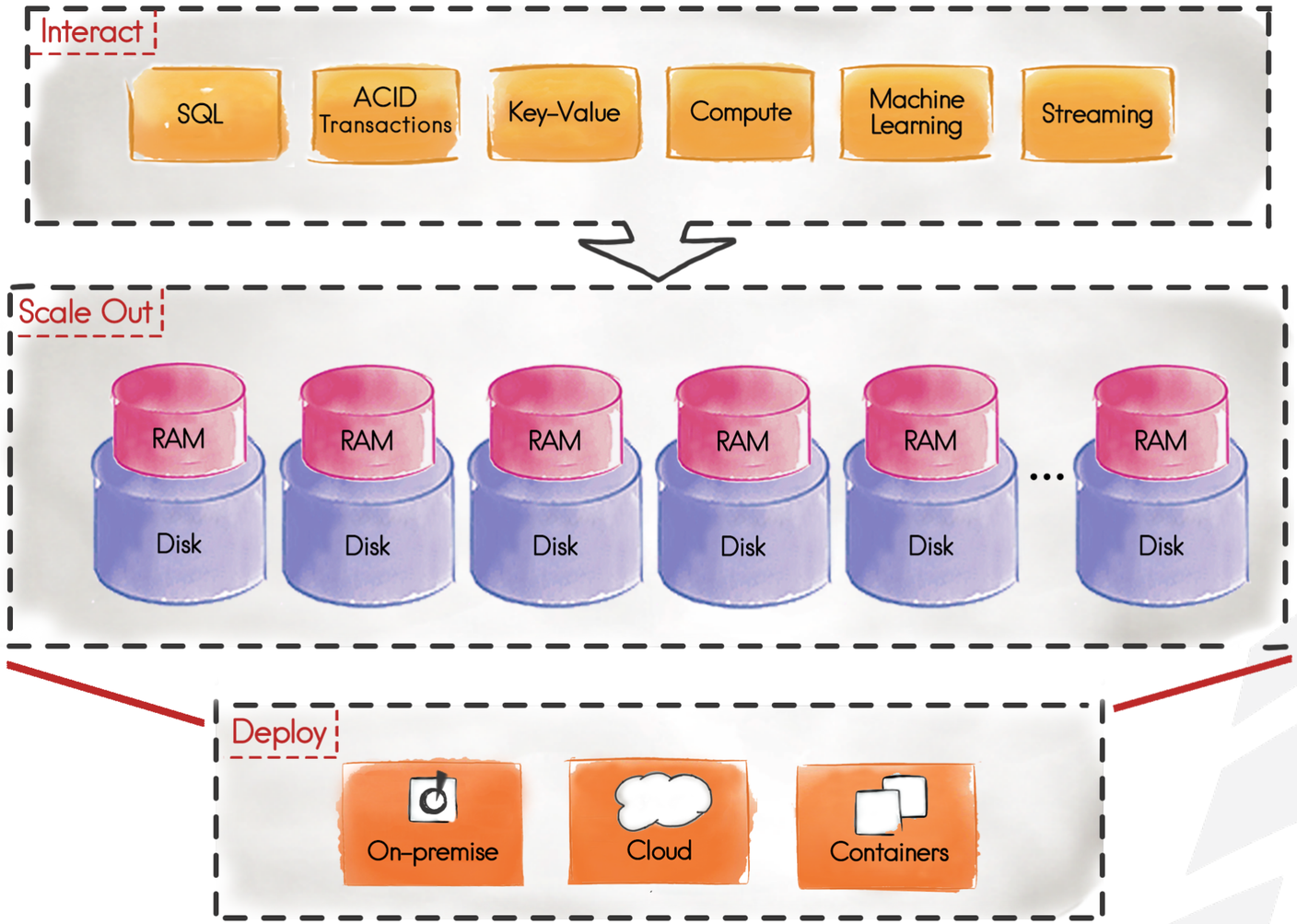
GridGain

Agenda

- Apache Ignite 2.0
- Ignite Virtual Memory
  - Architecture
  - Secondary Storage
- Distributed SQL Database
  - DML
  - DDL

Apache Ignite 2.0

Apache Ignite 2.x



Apache Ignite 2.0

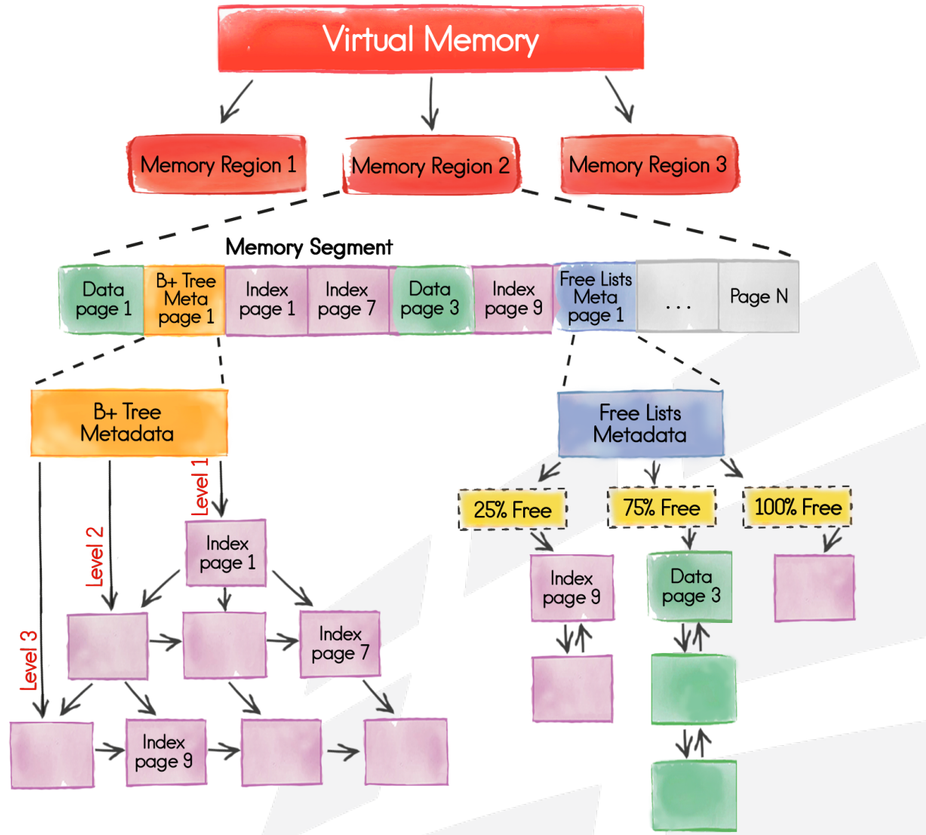
- Ignite Virtual Memory
- Data Definition Language
- Machine Learning Grid
- .NET Plugins System
- C++ Custom Code Execution
- Integrations
  - Spring Data Integration
  - Hibernate 5
  - Rocket MQ

Ignite Virtual Memory

Overview

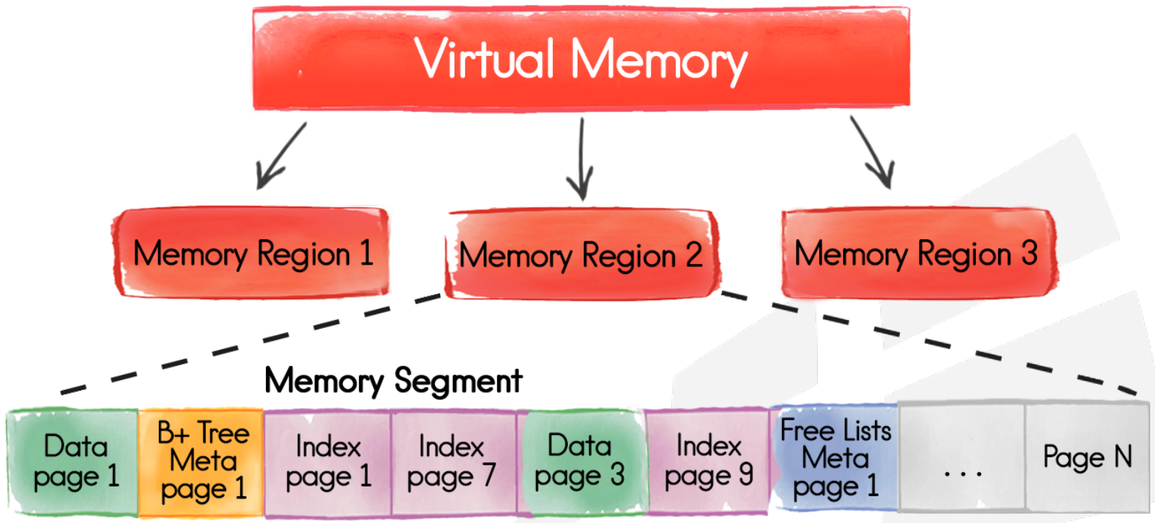
Ignite Virtual Memory: Benefits

- Off-heap page based memory
- Excludes noticeable GC pauses
- Predictable memory consumption
- Automatic defragmentation
- Seamless disk integration
- Boosts SQL performance



Ignite Virtual Memory

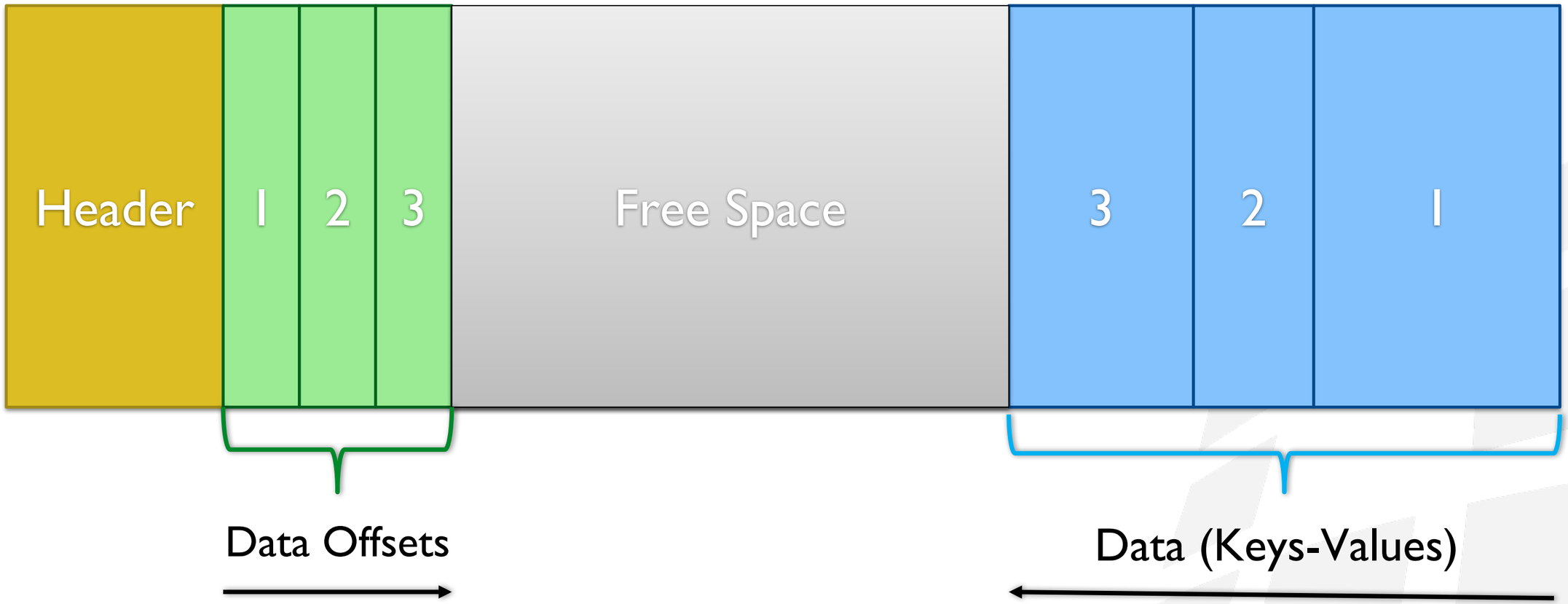
- Consumes Total Space Available
  - RAM and disk
  - Default Behavior
- Memory Region
  - Initial & max size
  - Data eviction mode
- Memory Segment
  - Continuous physical space
- Page
  - Fixed-length block



Pages

- Fixed-length block
- Frugal memory usage
- Automatic defragmentation
- Data Page
  - Stores key-value pairs
- Index Page
  - Linked by B+Tree
  - Refers to data pages
- Meta Page

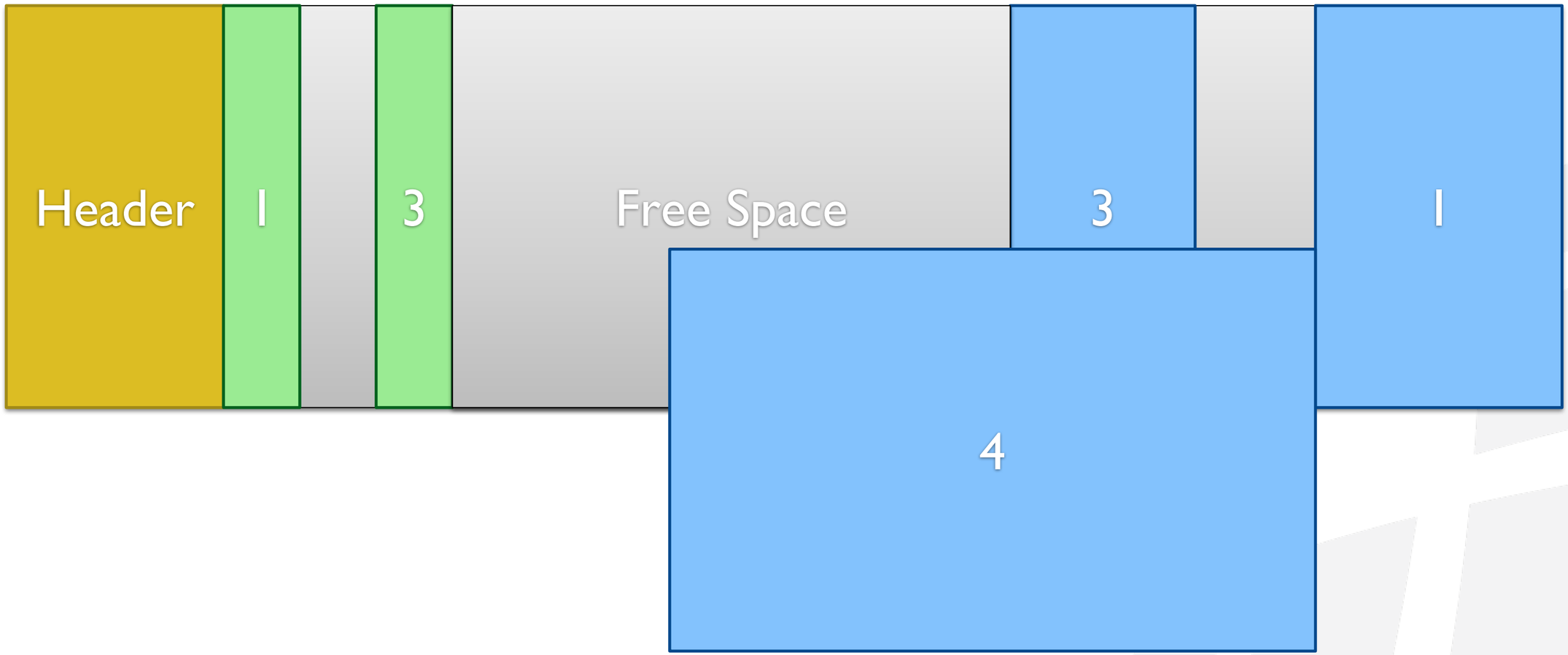
Data Page



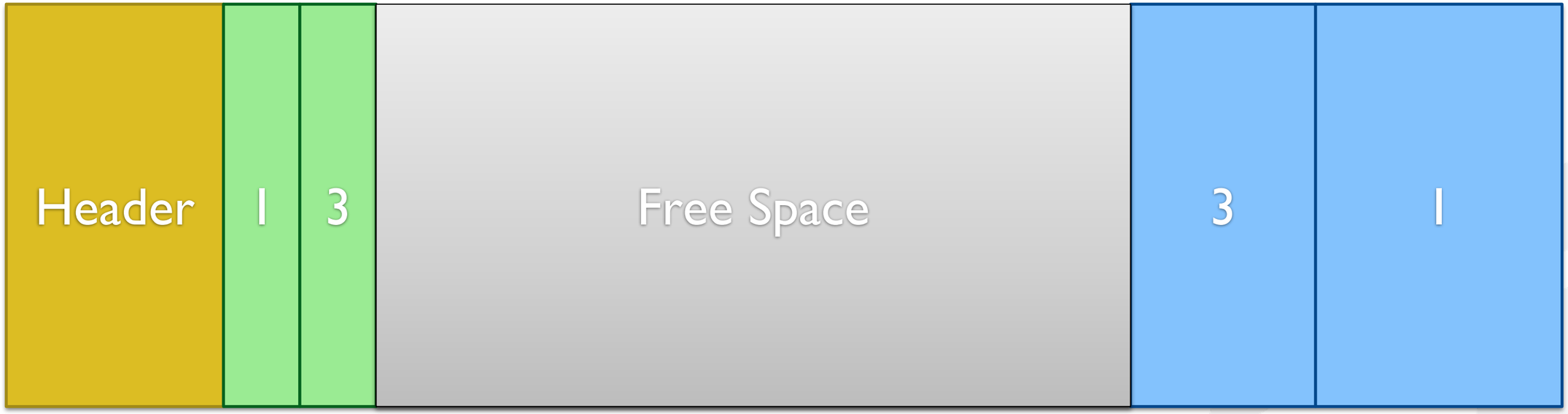
Defragmentation

Defragmentation

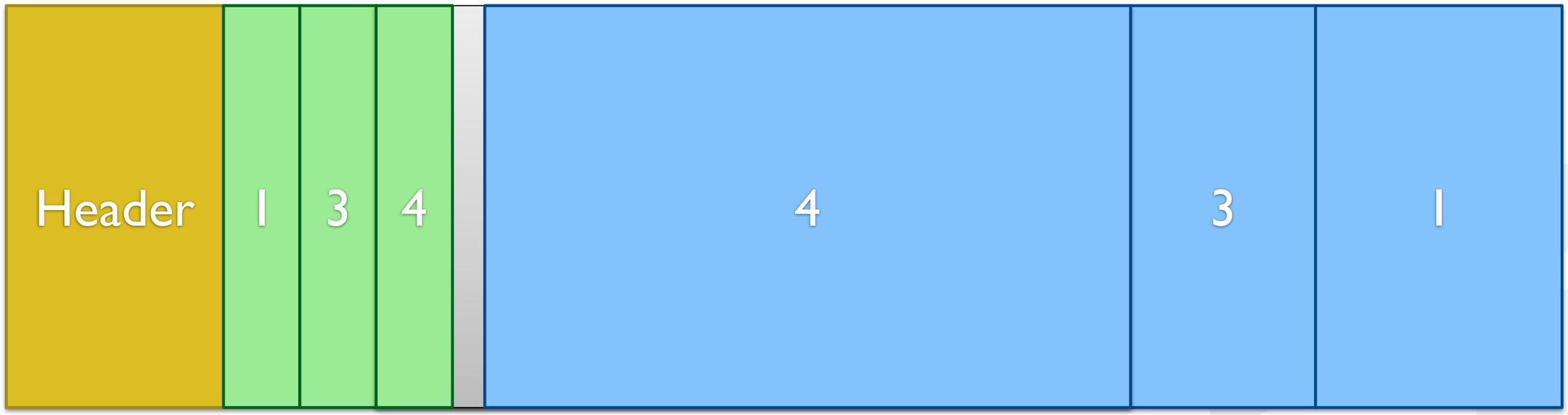
Defragmentation



Defragmentation



Defragmentation



Data Eviction

- Page-based Eviction
  - Random-LRU
  - Random-2-LRU
- On-heap Entries Cache
  - LRU
  - FIFO
  - Sorted

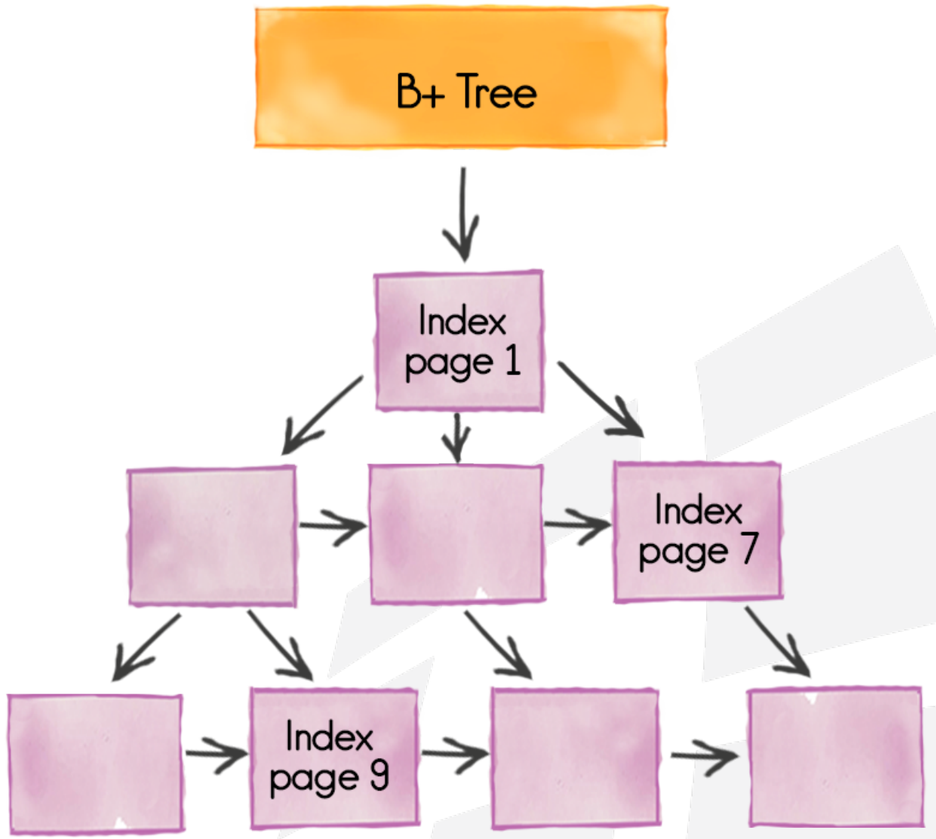
```
<bean class="org.apache.ignite.configuration.MemoryConfiguration">
  <!-- Defining additional memory poolicies. -->
  <property name="memoryPolicies">
    <list>
      <!--
          Defining a policy for 20 GB memory region with RANDOM_2_LRU eviction.
      -->
      <bean class="org.apache.ignite.configuration.MemoryPolicyConfiguration">
        <property name="name" value="20GB_Region_Eviction"/>
        <!-- Initial size is 5 GB. -->
        <property name="initialSize" value="#{5 * 1024 * 1024 * 1024}"/>
        <!-- Maximum size is 20 GB. -->
        <property name="maxSize" value="#{20 * 1024 * 1024 * 1024}"/>
        <!-- Enabling RANDOM_2_LRU eviction. -->
        <property name="pageEvictionMode" value="RANDOM_2_LRU"/>
      </bean>
    </list>
    ...
  </property>
  ...
</bean>
```

Data Eviction

```
<bean class="org.apache.ignite.configuration.MemoryConfiguration">
  <!-- Defining additional memory poolicies. -->
  <property name="memoryPolicies">
    <list>
      <!--
          Defining a policy for 20 GB memory region with RANDOM_2_LRU eviction.
      -->
      <bean class="org.apache.ignite.configuration.MemoryPolicyConfiguration">
        <property name="name" value="20GB_Region_Eviction"/>
        <!-- Initial size is 5 GB. -->
        <property name="initialSize" value="#{5 * 1024 * 1024 * 1024}"/>
        <!-- Maximum size is 20 GB. -->
        <property name="maxSize" value="#{20 * 1024 * 1024 * 1024}"/>
        <!-- Enabling RANDOM_2_LRU eviction. -->
        <property name="pageEvictionMode" value="RANDOM_2_LRU"/>
      </bean>
    </list>
    ...
  </property>
  ...
</bean>
```

B+Tree

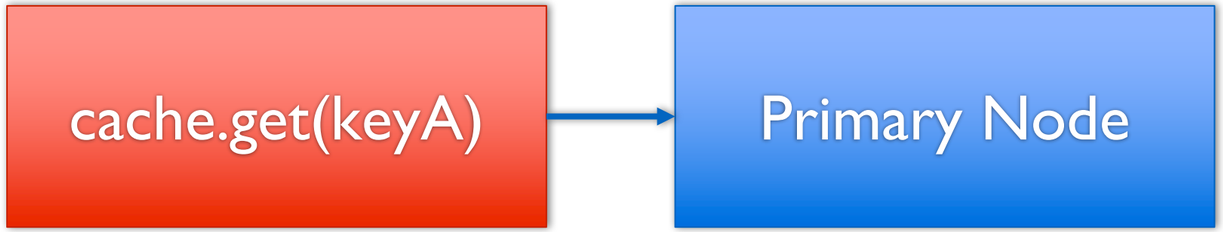
- Self-balancing Tree
  - Memory & Disk
- Links and Sorts Index Pages
- Sorted Index
  - Custom indexes
- Hash Index
  - Primary Keys
  - Hash code based sorting



B+Tree: Key-Value Read

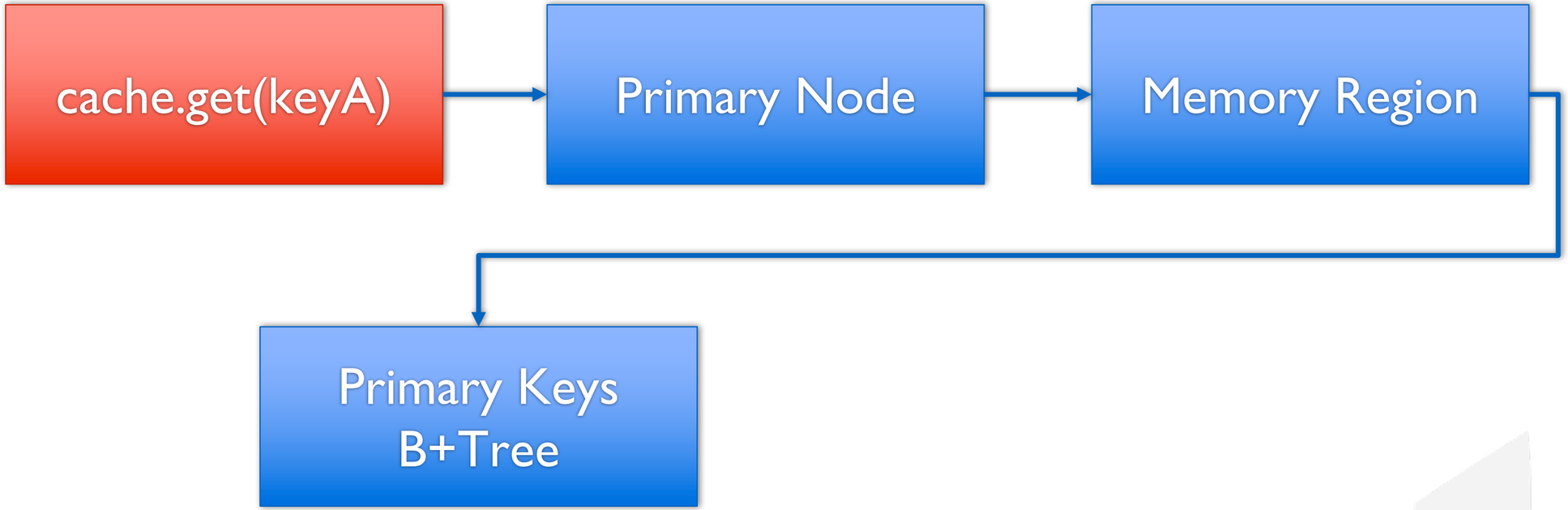
cache.get(keyA)

B+Tree: Key-Value Read

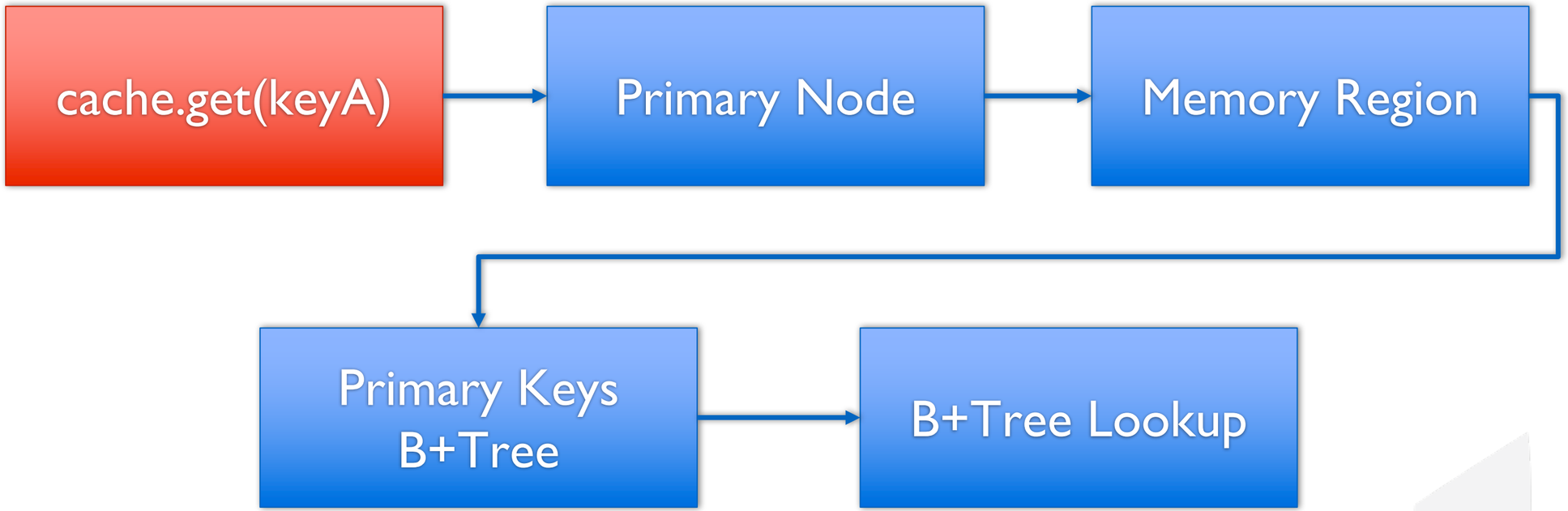


B+Tree: Key-Value Read

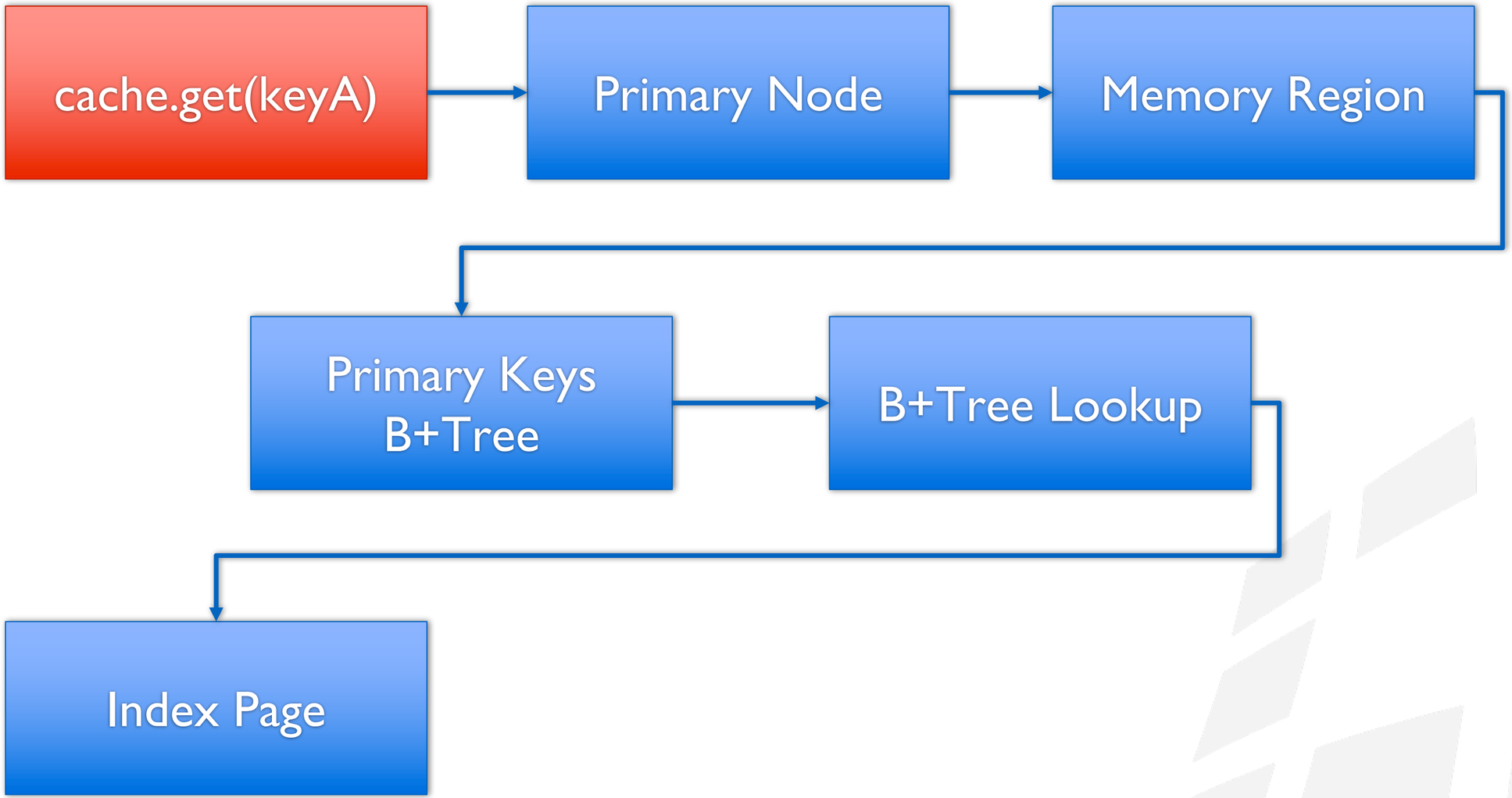
B+Tree: Key-Value Read



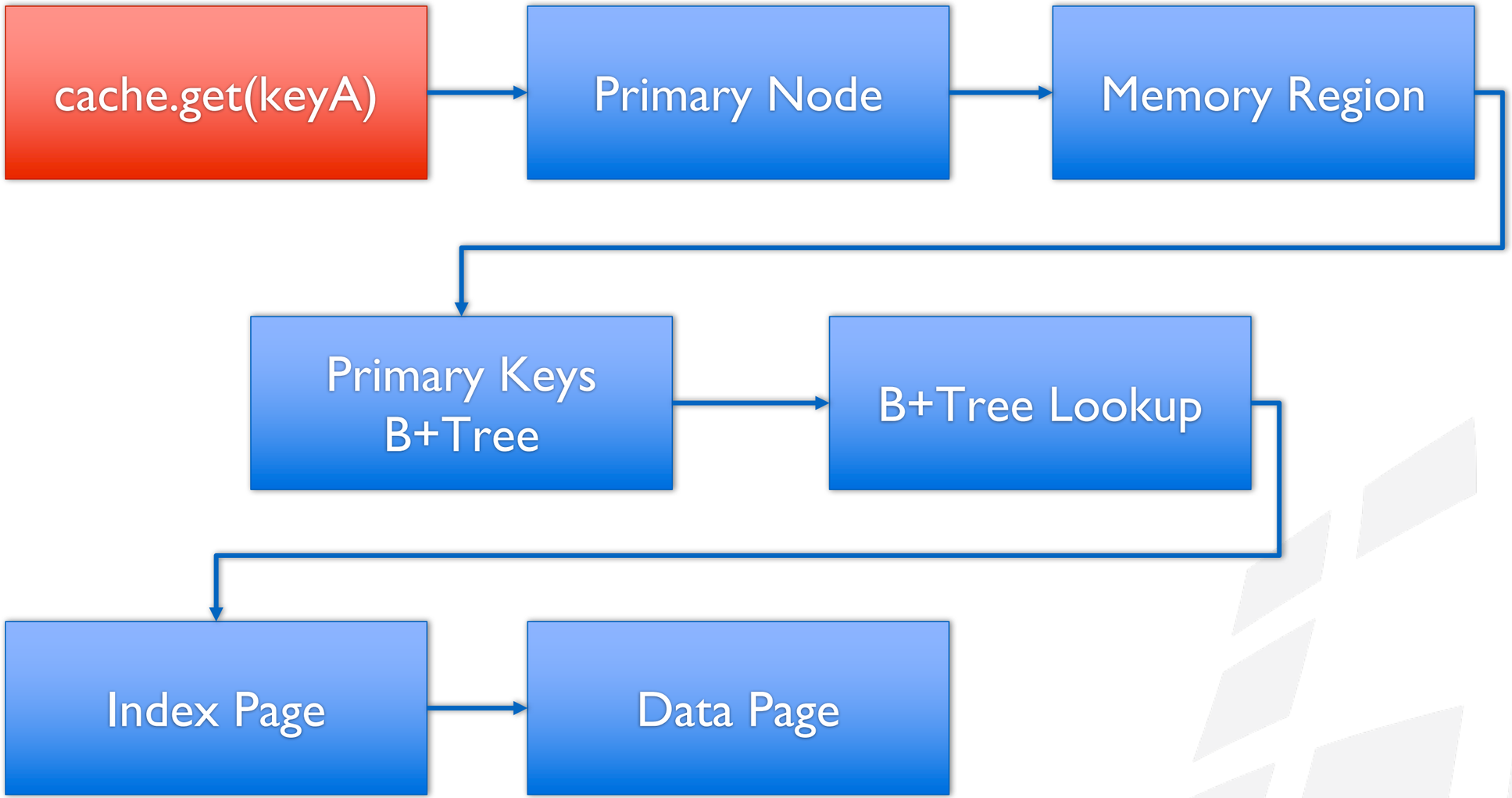
B+Tree: Key-Value Read



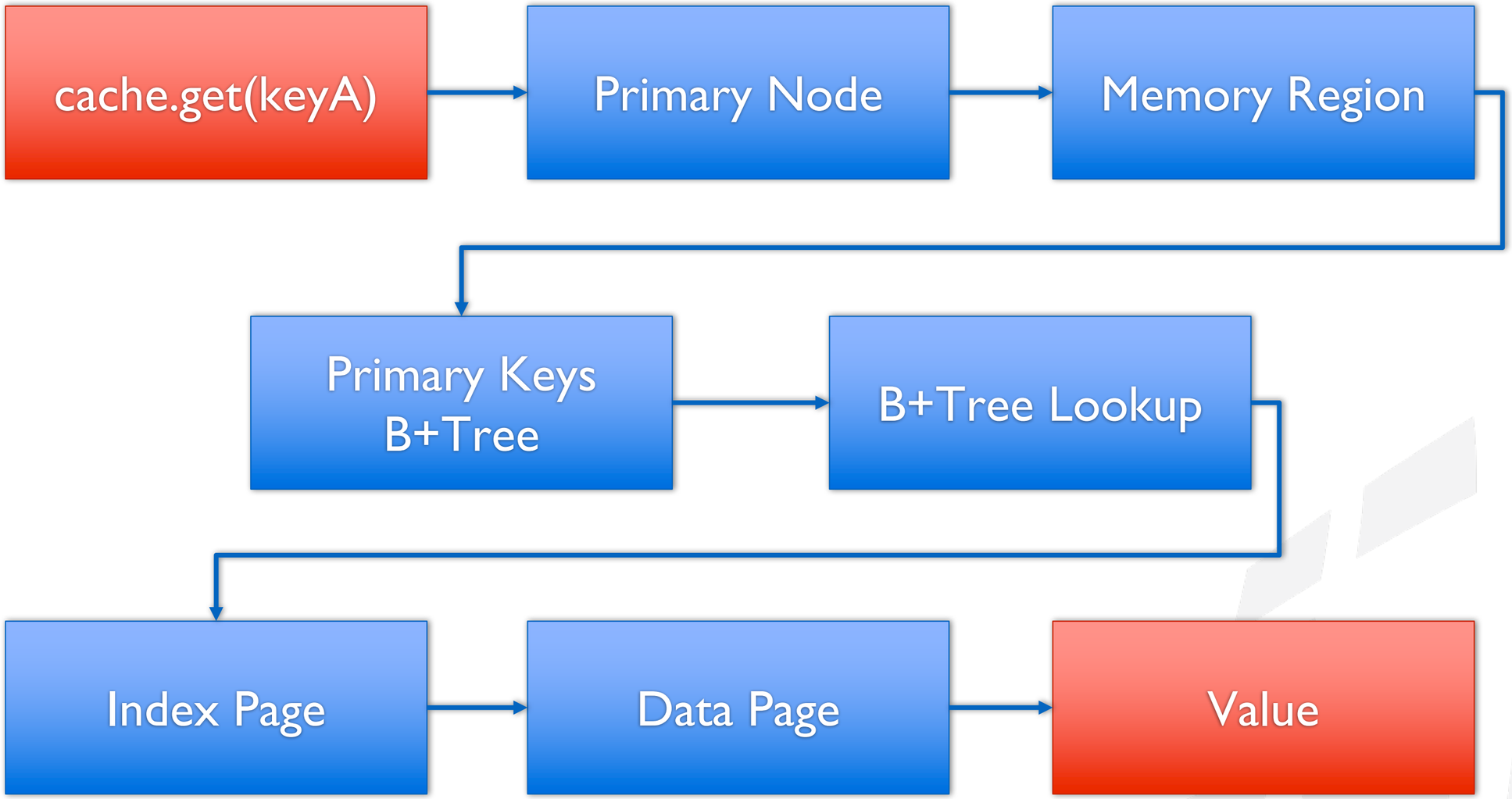
B+Tree: Key-Value Read



B+Tree: Key-Value Read

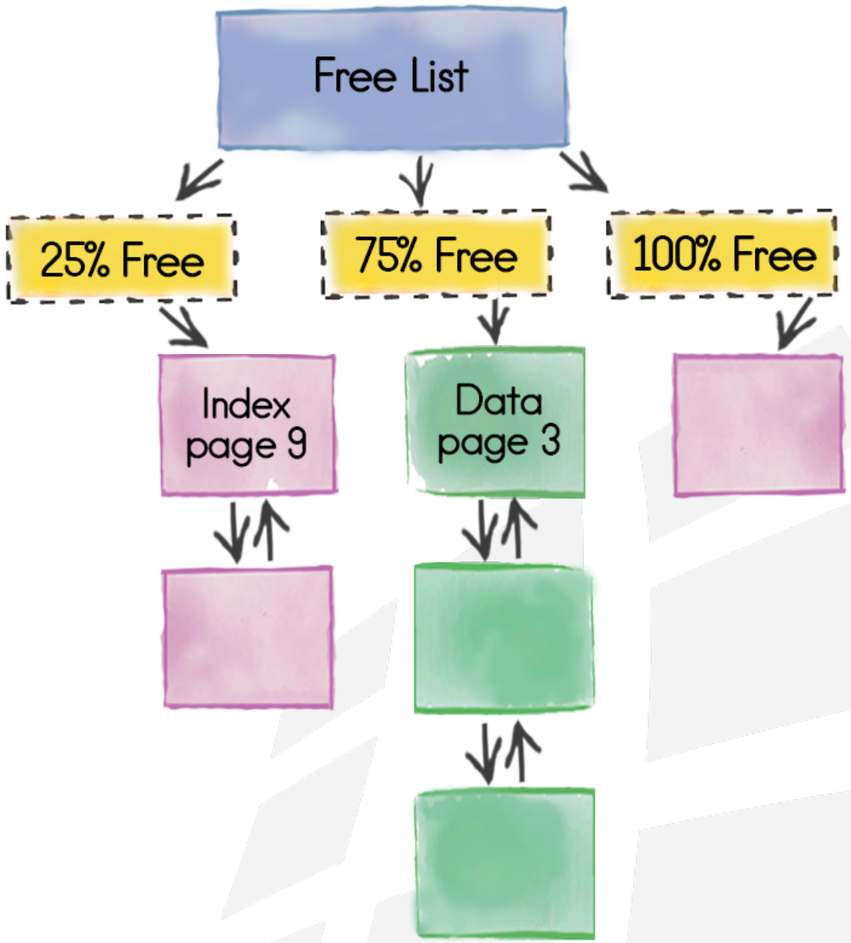


B+Tree: Key-Value Read



Free Lists

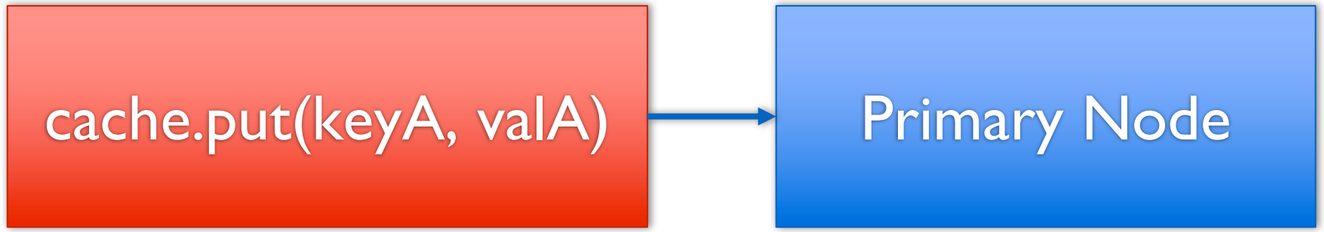
- Tracks Pages of ~ Equal Free Space
  - 25% free
  - 75% free
  - etc.
- Essential for Update Operations
  - Returns a page with min space needed
  - Reduces fragmentation
  - Lowers compaction activity



B+Tree and Free Lists: Updates

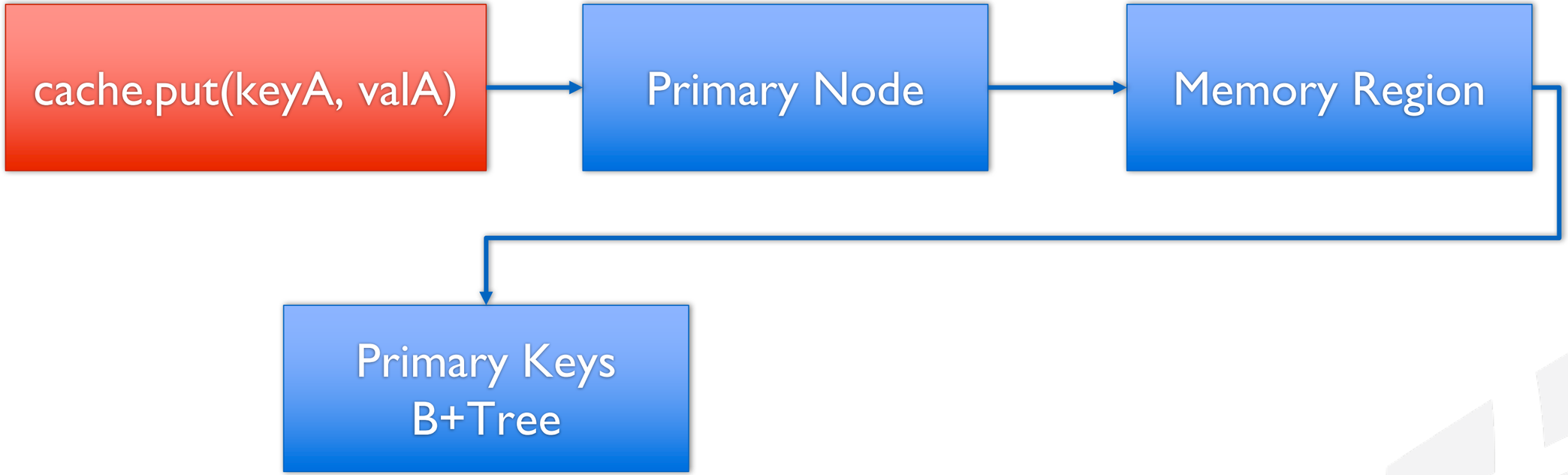
cache.put(keyA, valA)

B+Tree and Free Lists: Updates

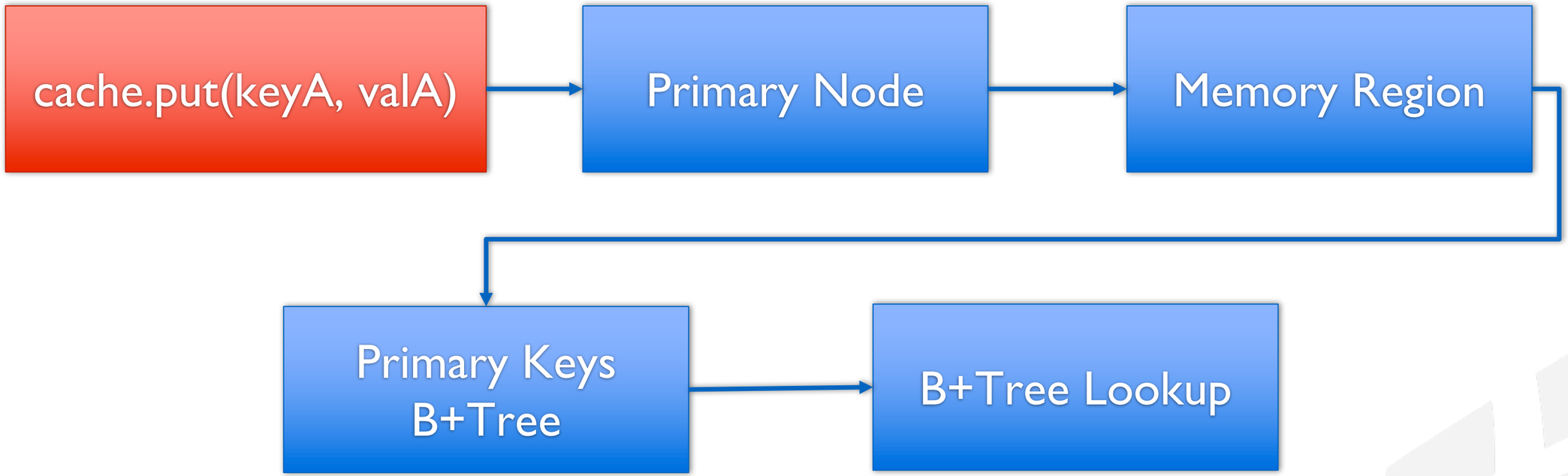


B+Tree and Free Lists: Updates

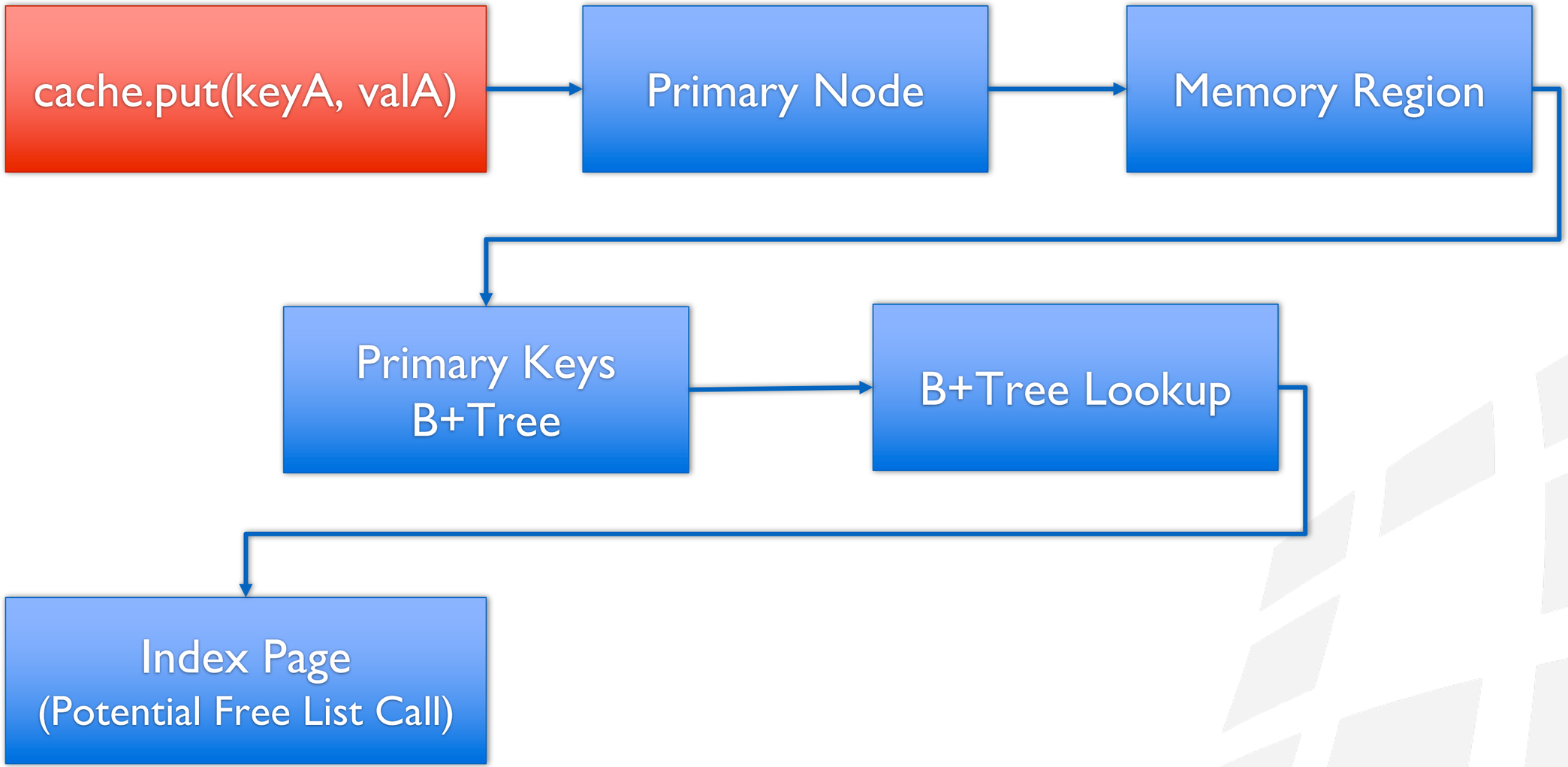
B+Tree and Free Lists: Updates



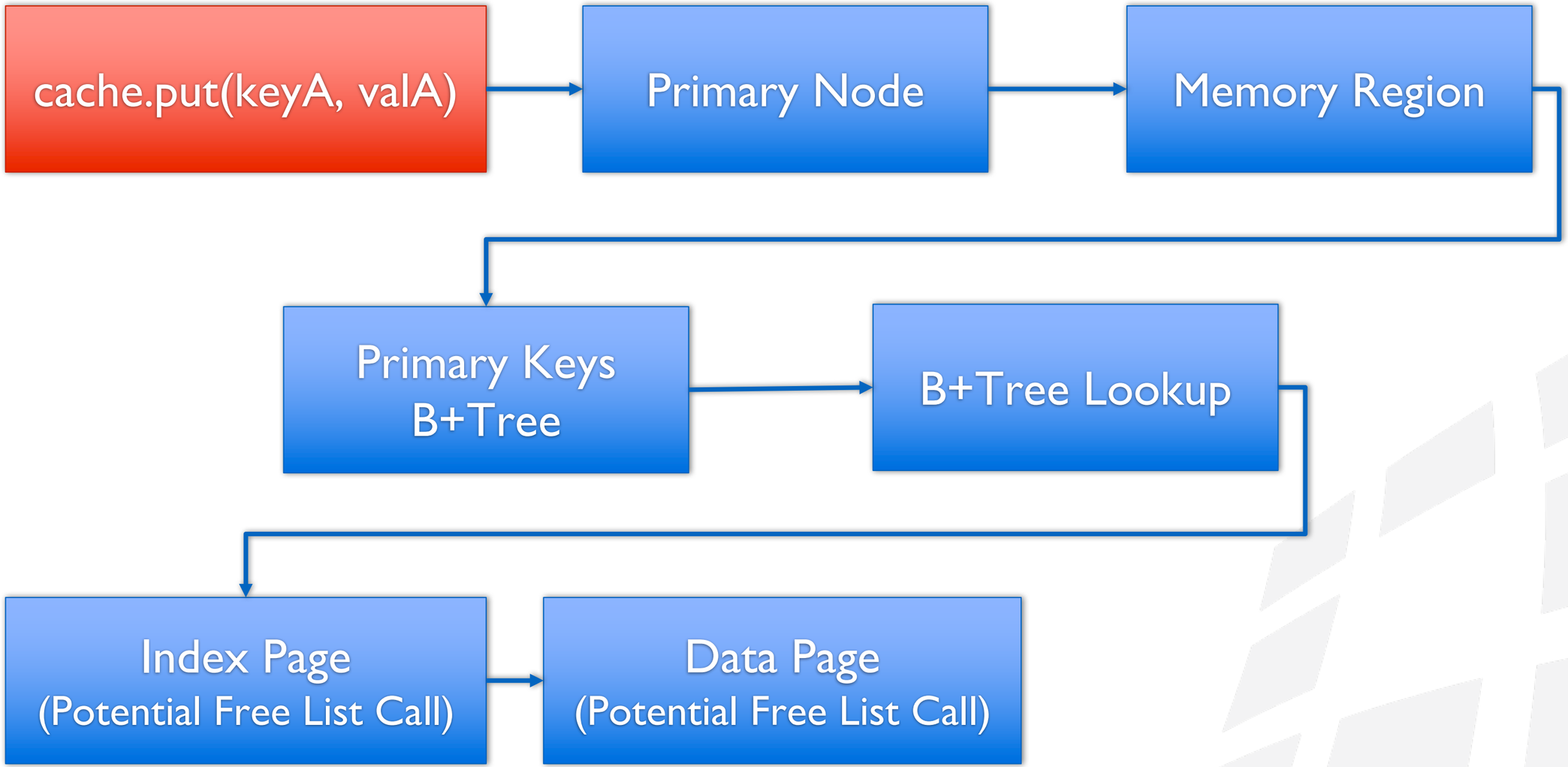
B+Tree and Free Lists: Updates



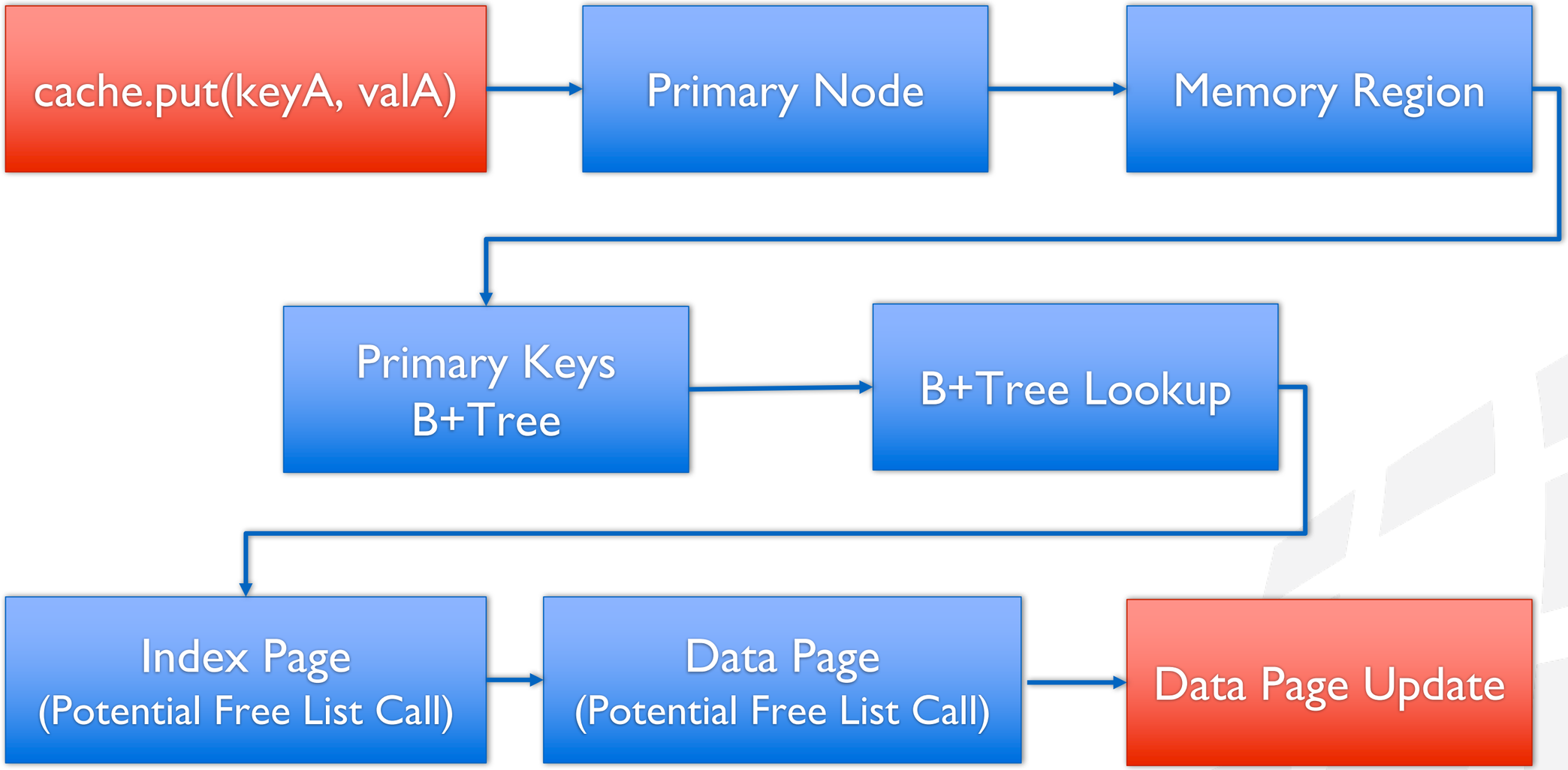
B+Tree and Free Lists: Updates



B+Tree and Free Lists: Updates



B+Tree and Free Lists: Updates

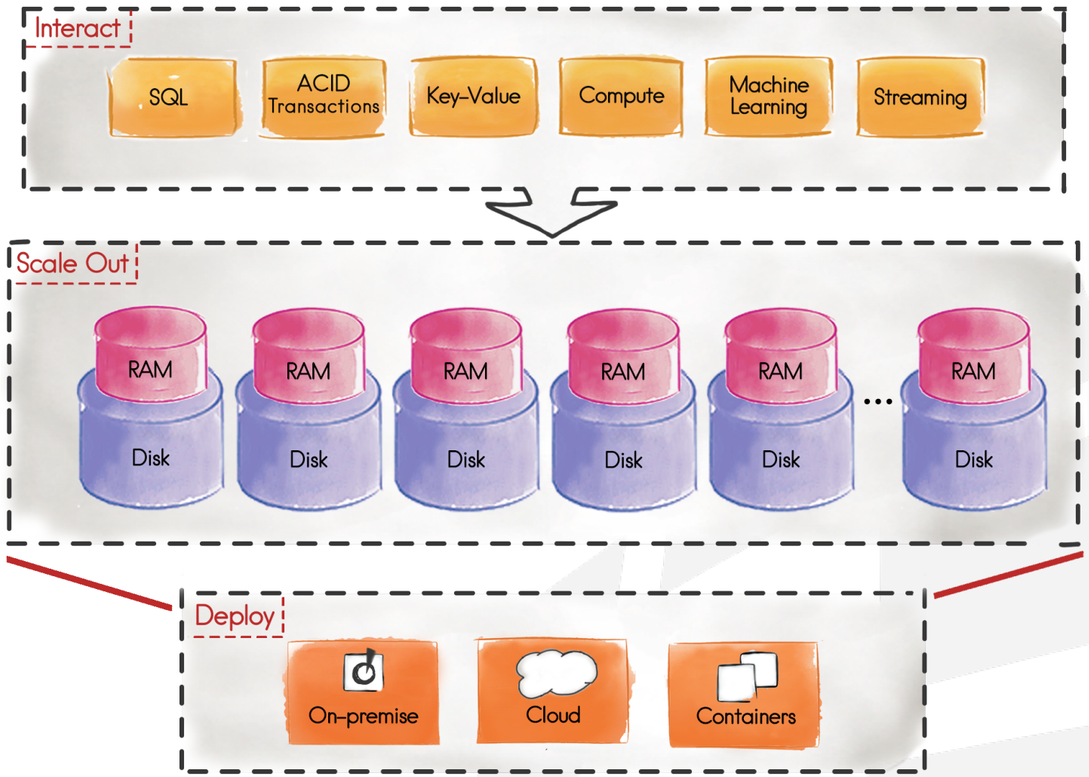


Virtual Page Memory

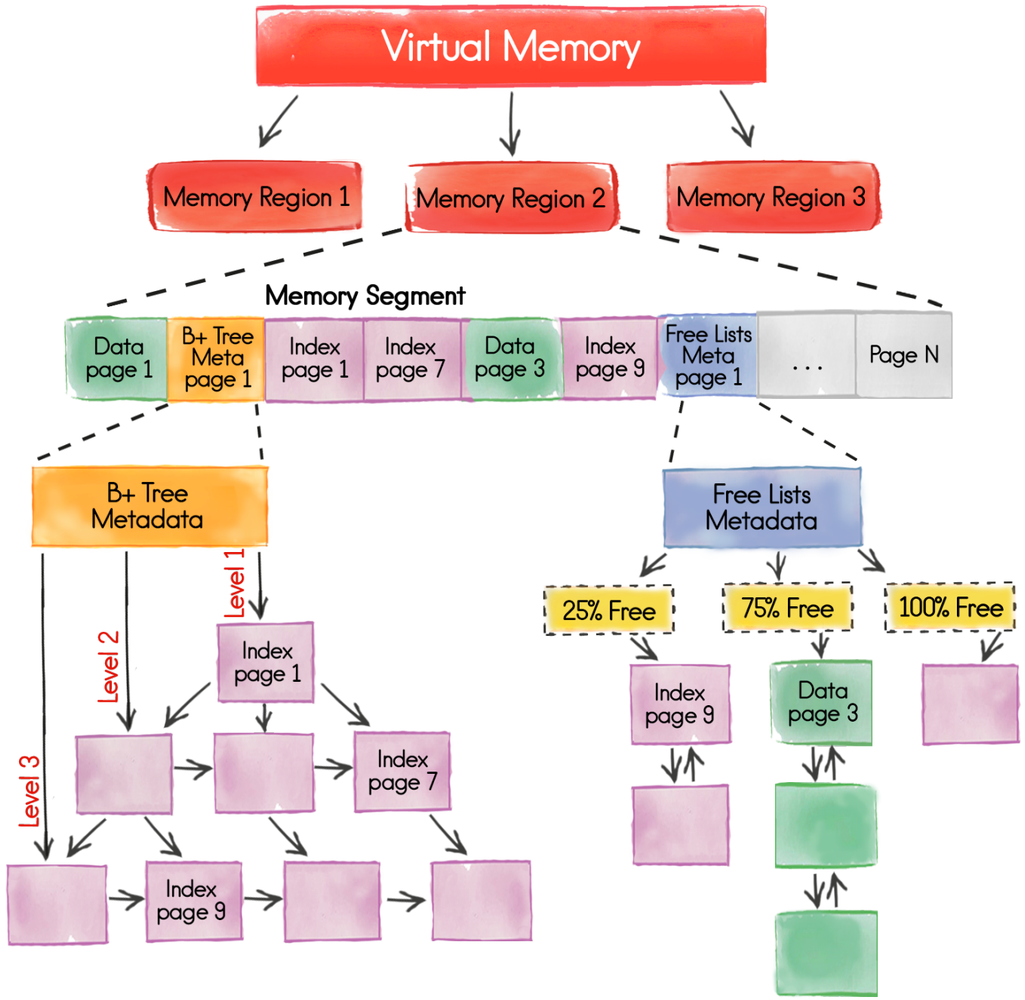
Secondary Storage

Persistent Store

- Expands Virtual Memory to Disk
  - Flash, SSD, Intel 3D Xpoint
- ANSI-99 SQL Compliant
  - Join RAM and disk datasets
- ACID Compliant
  - Write-ahead logs
- Stores superset of data
  - If a page is in RAM it's always on disk
  - Same format
- Instantaneous Restarts
- Expect in Apache Ignite 2.x



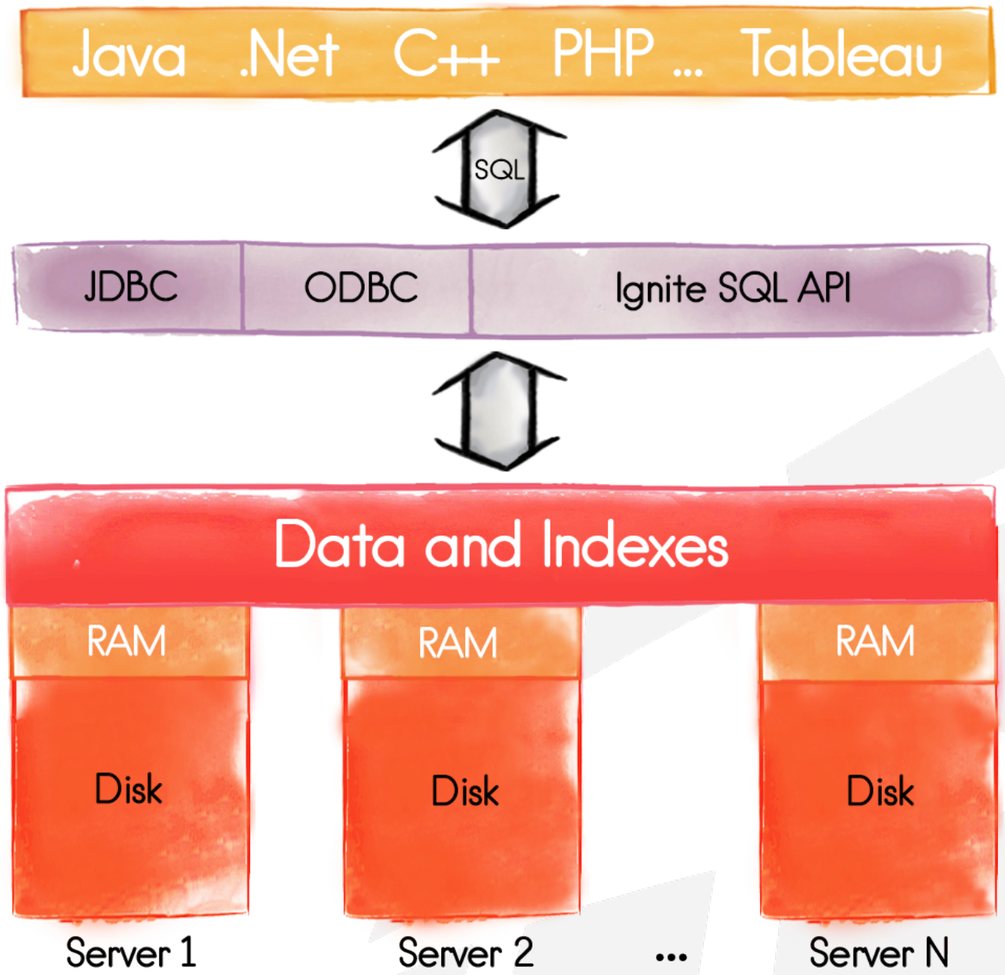
Ignite Virtual Memory



Distributed SQL Database

Ignite Distributed SQL Database

- Distributed SQL Database
  - RAM and disk
- ANSI-99 and ACID Compliant
- Highly Available
- Scales to 1000s of nodes
- Deploys on premise and in cloud
- Cross-platform
  - ODBC
  - JDBC

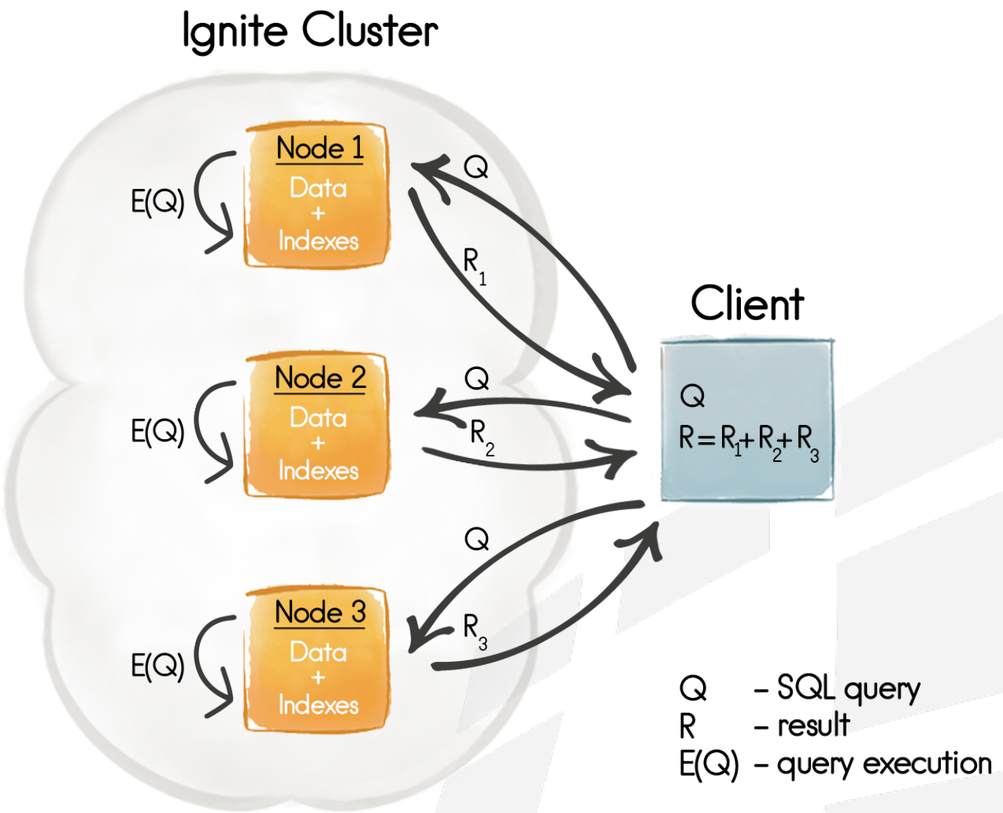


Distributed SQL Database

Data Modification Language

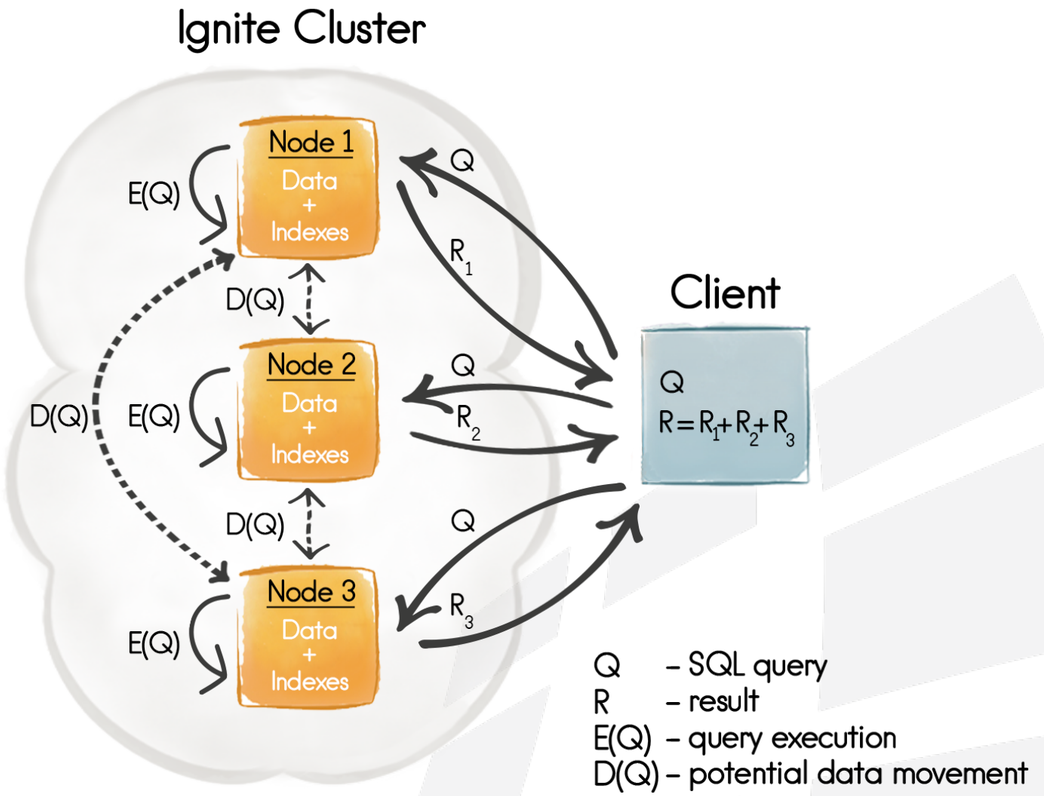
SQL Queries

- Distributed Joins
- Automatic Group By, Aggregations, Sorting
- Ad-Hoc SQL Support
- Off-heap and disk indexes



Non-Collocated Joins

- Non-Collocated Mode
  - No need to collocate data
  - Potential data movement between nodes
- Use case
  - No feasible to collocate data for particular SQL queries
- Disabled by default



Coding Examples

```java
IgniteCache<AffinityKey<UUID>, Person> cache = ignite.cache("persons");

// Query to get salaries grouped by organization.
SqlFieldsQuery qry = new SqlFieldsQuery(
    "select org.name, avg(salary), max(salary), min(salary) " +
        "from Person, \"Organizations\".Organization as org " +
        "where Person.orgId = org.id " +
        "group by org.name " +
        "order by org.name");

QueryCursor<List<?>> cursor = cache.query(qry);

List<List<?>> res = cursor.getAll();
```

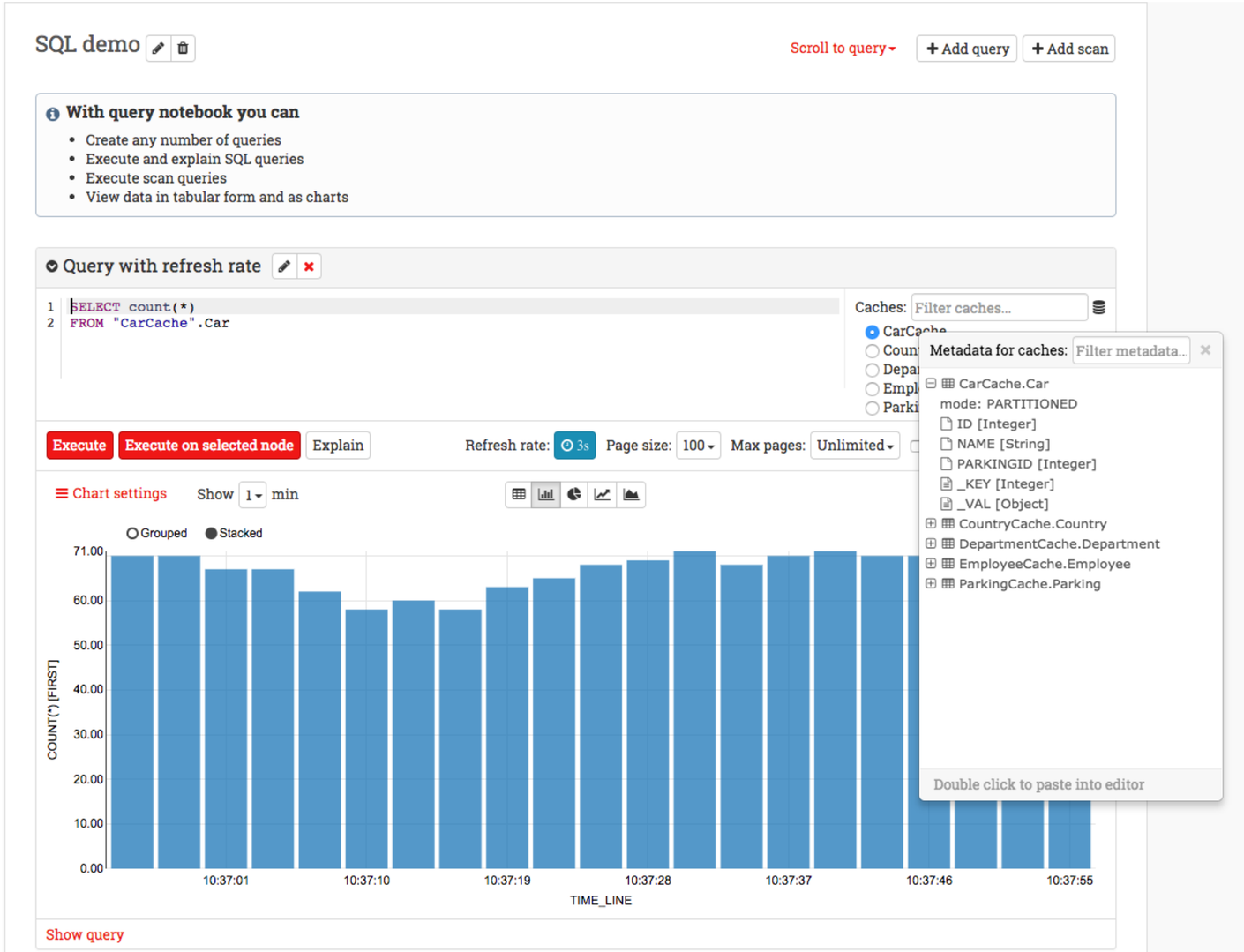

```php
// Preparing query.
$dbs = $dbh->prepare(
  'INSERT INTO Person (_key, firstName, lastName, resume, salary)
   VALUES (?, ?, ?, ?, ?)');

// Declaring parameters.
$key = 777;
$firstName = "James";
$lastName = "Bond";
$resume = "Secret Service agent";
$salary = 65000;

// Binding parameters.
$dbs->bindParam(1, $key);
$dbs->bindParam(2, $firstName);
$dbs->bindParam(3, $lastName);
$dbs->bindParam(4, $resume);
$dbs->bindParam(5, $salary);

// Executing the query.
$dbs->execute();
```

Data Modification

- INSERT
- UPDATE
- DELETE
- MERGE
- APIs
  - ODBC & JDBC
  - Java, .NET and C++

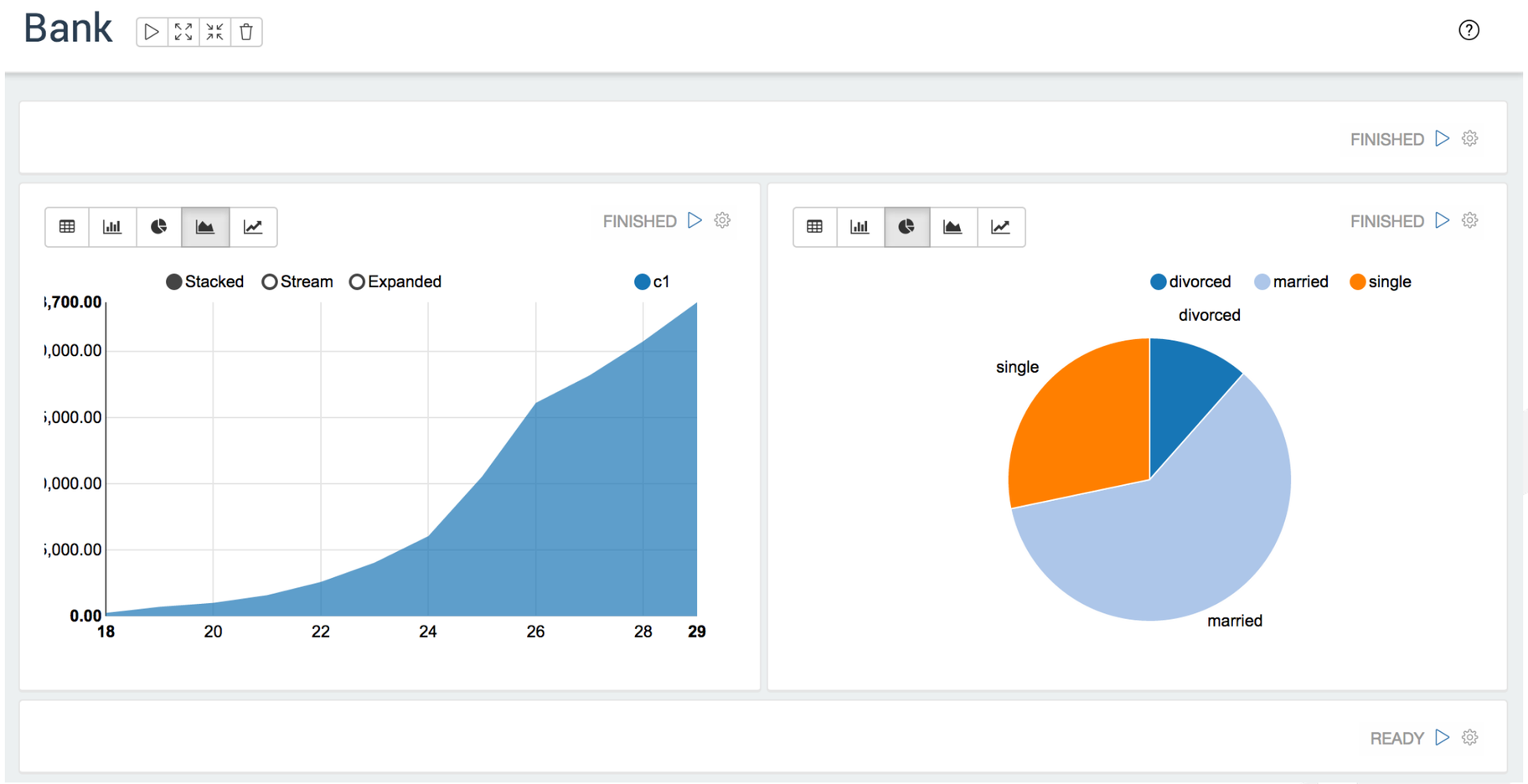
```
void AdjustSalary(SQLHDBC dbc, int64_t key, double salary)
{
  SQLHSTMT stmt;

  // Allocate a statement handle
  SQLAllocHandle(SQL_HANDLE_STMT, dbc, &stmt);

  SQLCHAR query[] = "UPDATE Person SET salary=? WHERE _key=?";

  SQLBindParameter(stmt, 1, SQL_PARAM_INPUT,
      SQL_C_DOUBLE, SQL_DOUBLE, 0, 0, &salary, 0, 0);

  SQLBindParameter(stmt, 2, SQL_PARAM_INPUT, SQL_C_SLONG,
      SQL_BIGINT, 0, 0, &key, 0, 0);

  SQLExecDirect(stmt, query, static_cast<SQLSMALLINT>(sizeof(query)));

  // Releasing statement handle.
  SQLFreeHandle(SQL_HANDLE_STMT, stmt);
}
```

Apache Ignite Web Console



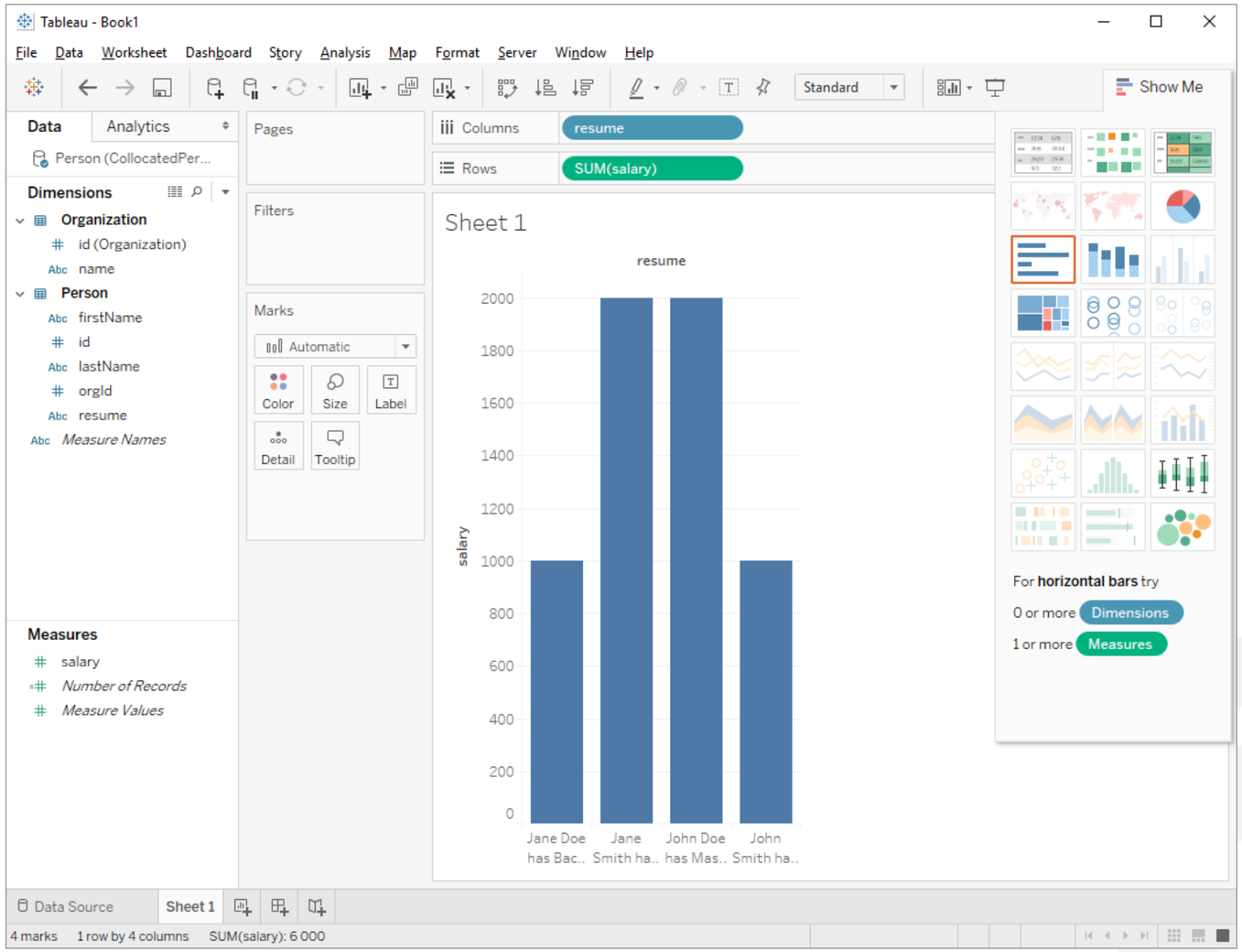
GridGain

Apache Zeppelin



GridGain

Data Analysis with Tableau



GridGain

Distributed SQL Database

Data Definition Language

Data Definition Language

- Create and drop indexes
  - In runtime and cluster-wide!
- Create and drop SQL Schema
  - Apache Ignite 2.1
- Use Ignite as SQL Database
  - No XML
  - No Java or .NET configuration
  - Connect, define, interact!

ANY QUESTIONS?

Thank you for joining us. Follow the conversation.

http://ignite.apache.org

 #apacheignite

GridGain