Couchbase

Couchbase Data Platform Overview

The Modern Database for Enterprise Applications

Agenda

1 Architecture Overview

2 Couchbase Services

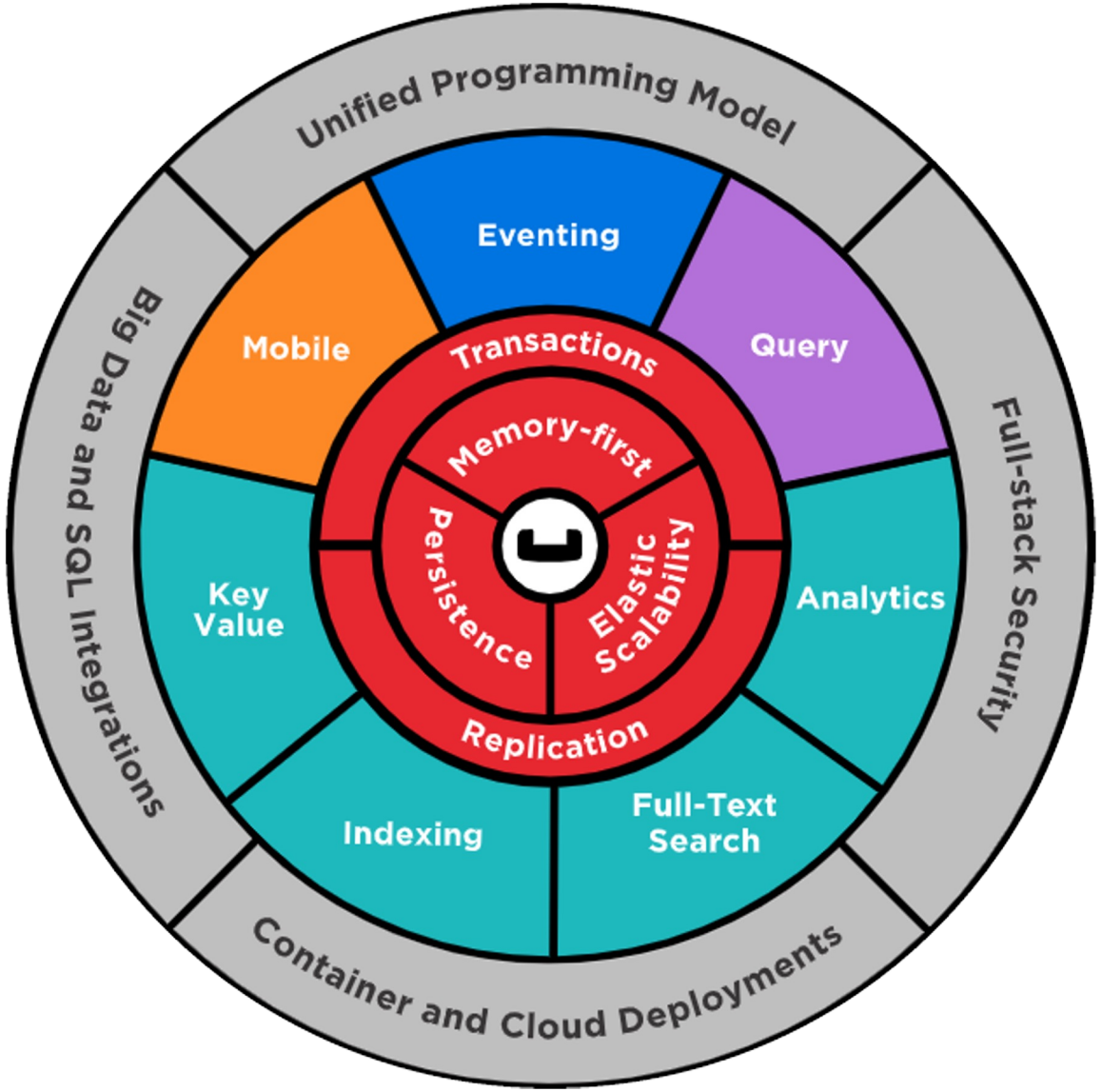
3 Core Engine Design

4 Use Cases @ Amadeus

1 Architecture Overview

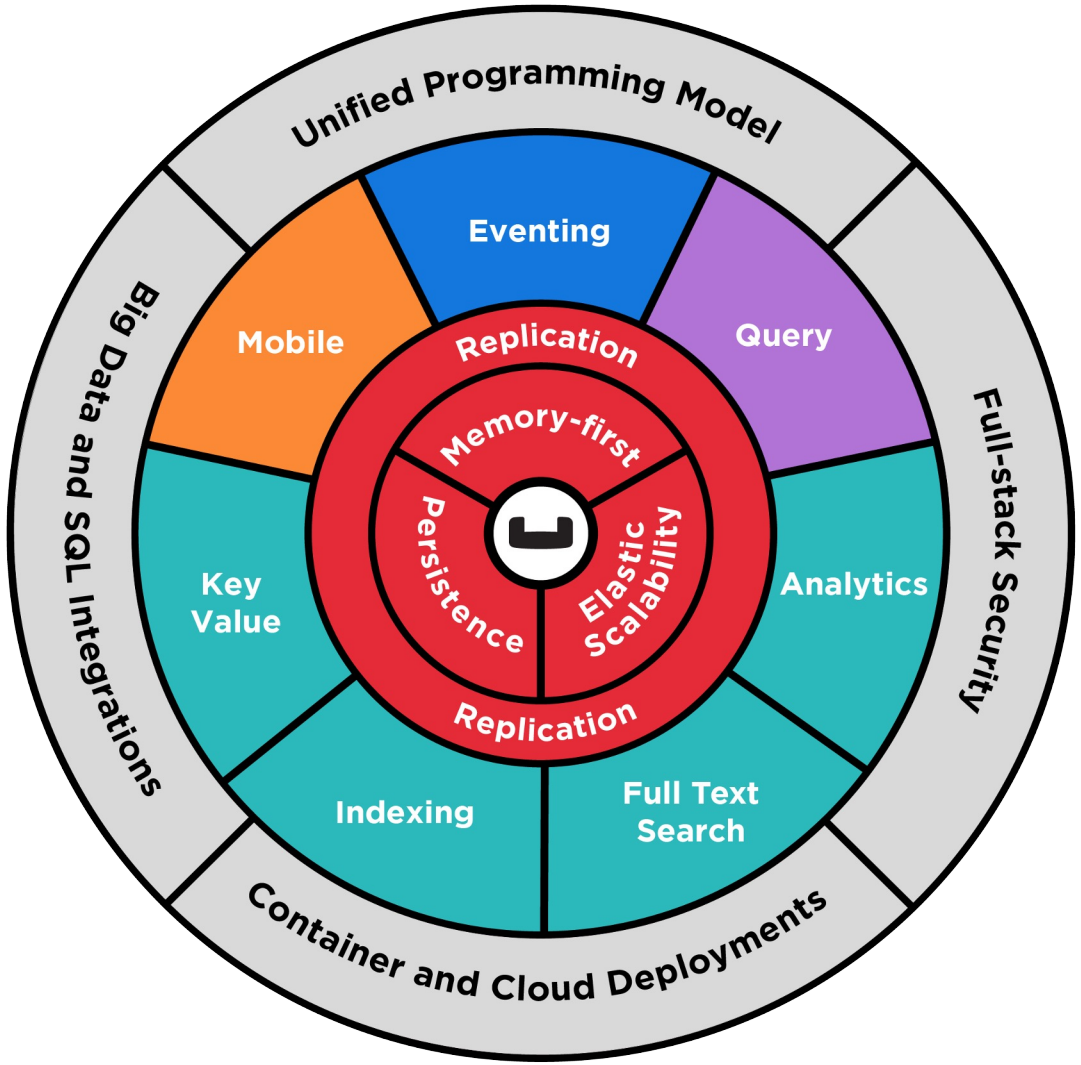
COUCHBASE ARCHITECTURE

- Unified Programming Interface
- Single Node Deployment
- Multi Dimensional Scaling
- Memory-first architecture
- High Availability
- Big data and SQL integrations
- Full-stack security
- Container and Cloud deployments



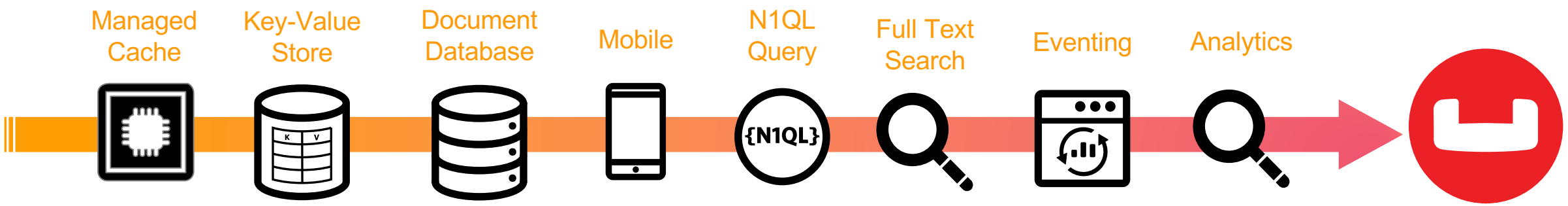
Couchbase Data Platform

- Memory-first
- Multi-service shared nothing architecture
- SQL-like Query Engine for JSON
- Clustered Global Indexes
- Active-Active Inter-DC Replication
- Full-Text Search
- Operational Analytics
- Stateless compute via Eventing
- End to End Security

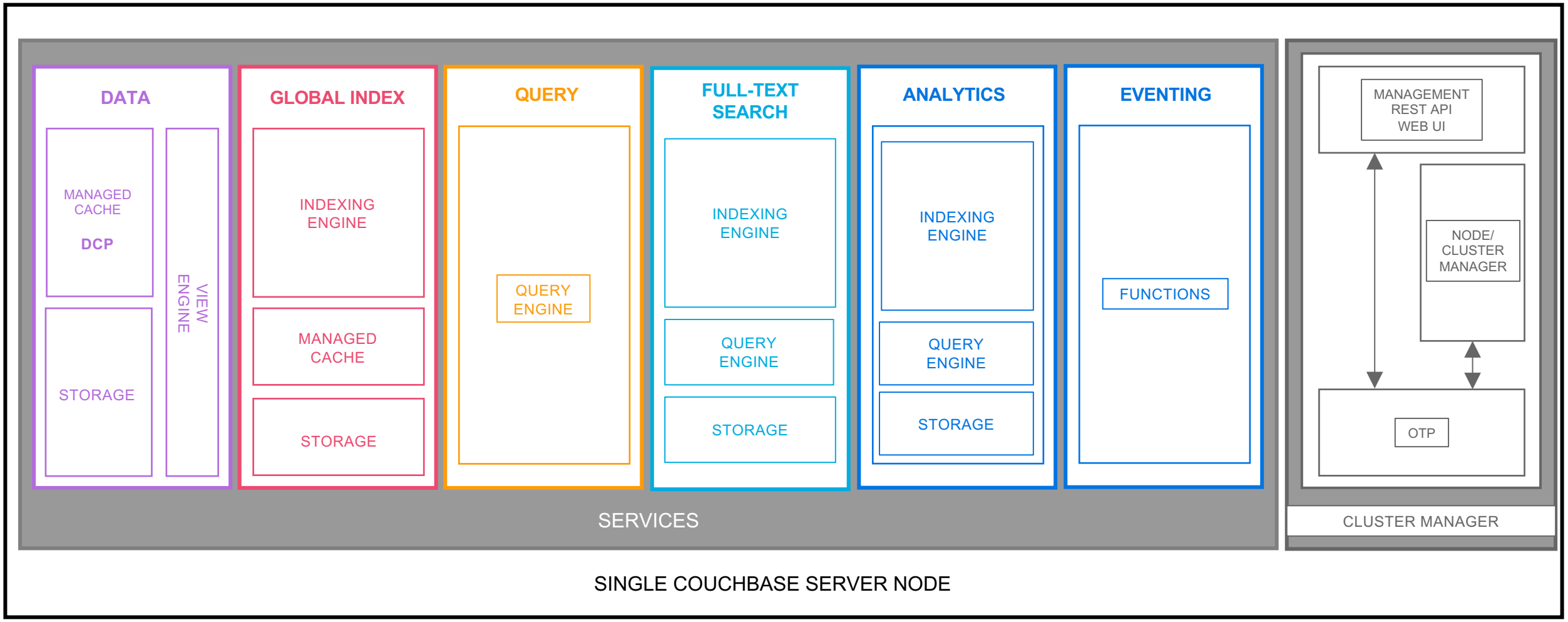


Couchbase: Core Principles Driving Evolution

- True auto sharding
- JSON-based flexible data model
- Memory-first Architecture
- Asynchronous everything
- Scale workloads independently



Couchbase Node Architecture

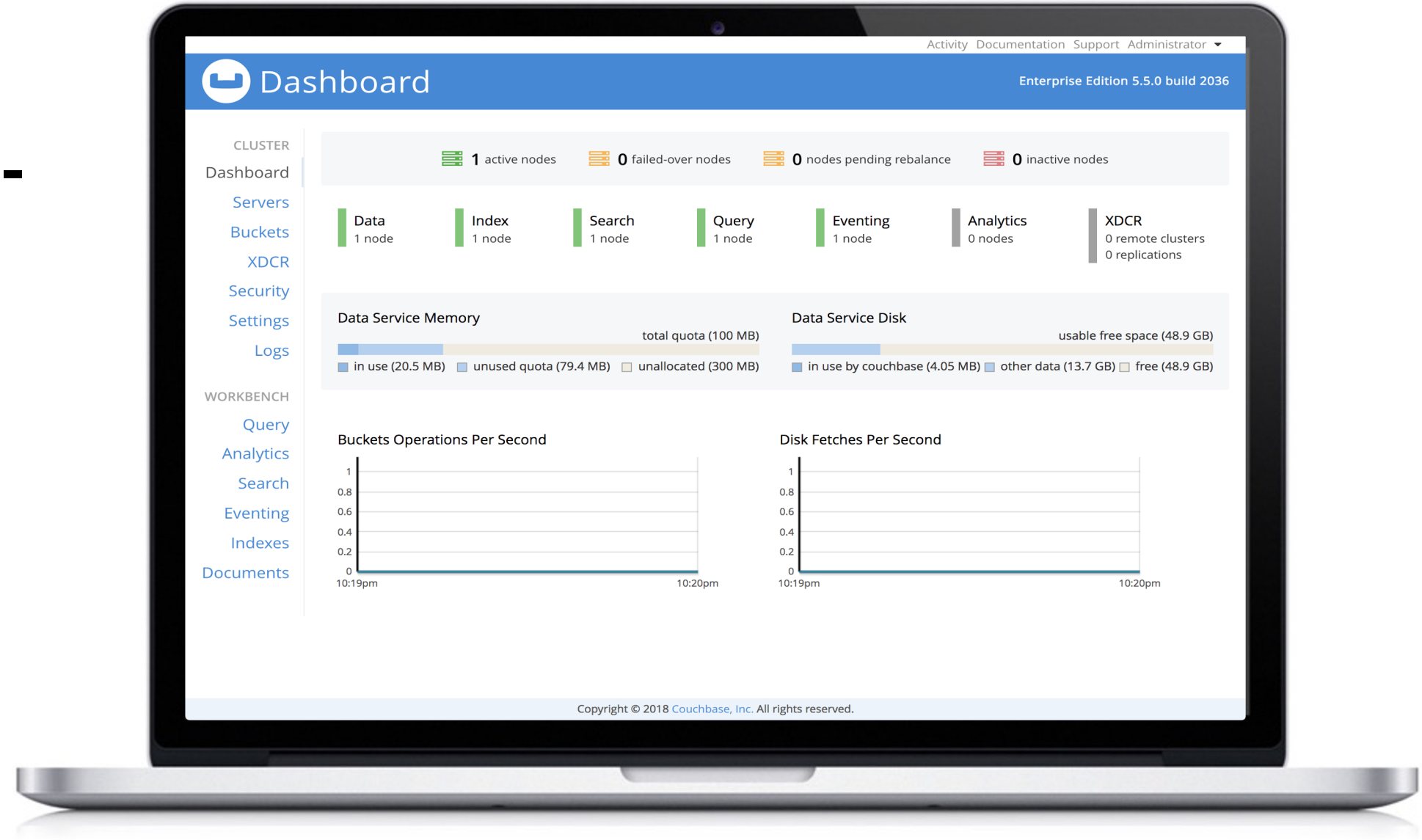


SINGLE COUCHBASE SERVER NODE

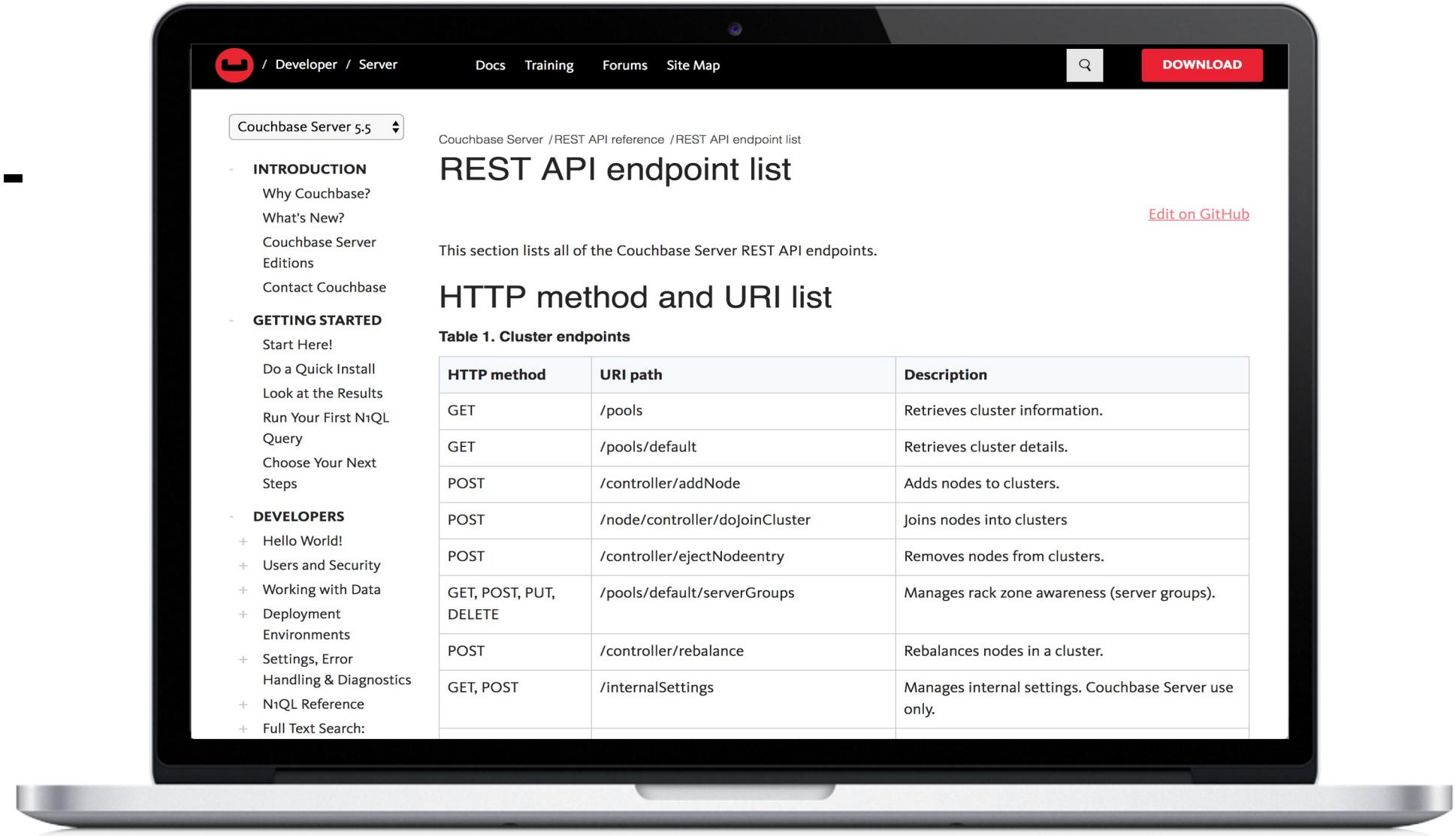
Cluster Manager

- Service-Centric Clustered Data System
  - Multi-process Architecture
  - Dynamic Distribution of Facilities
  - Cluster Map Client Distribution
  - Automatic Failover
- Security
  - Role-Based Administration
  - Audit Logging
- Enterprise Monitoring/Management
  - Full Graphical Web UI
  - REST API
  - Detailed Performance Stats
  - Automated Log Collection
  - Administrative Alerting

Cluster Manager - Web UI



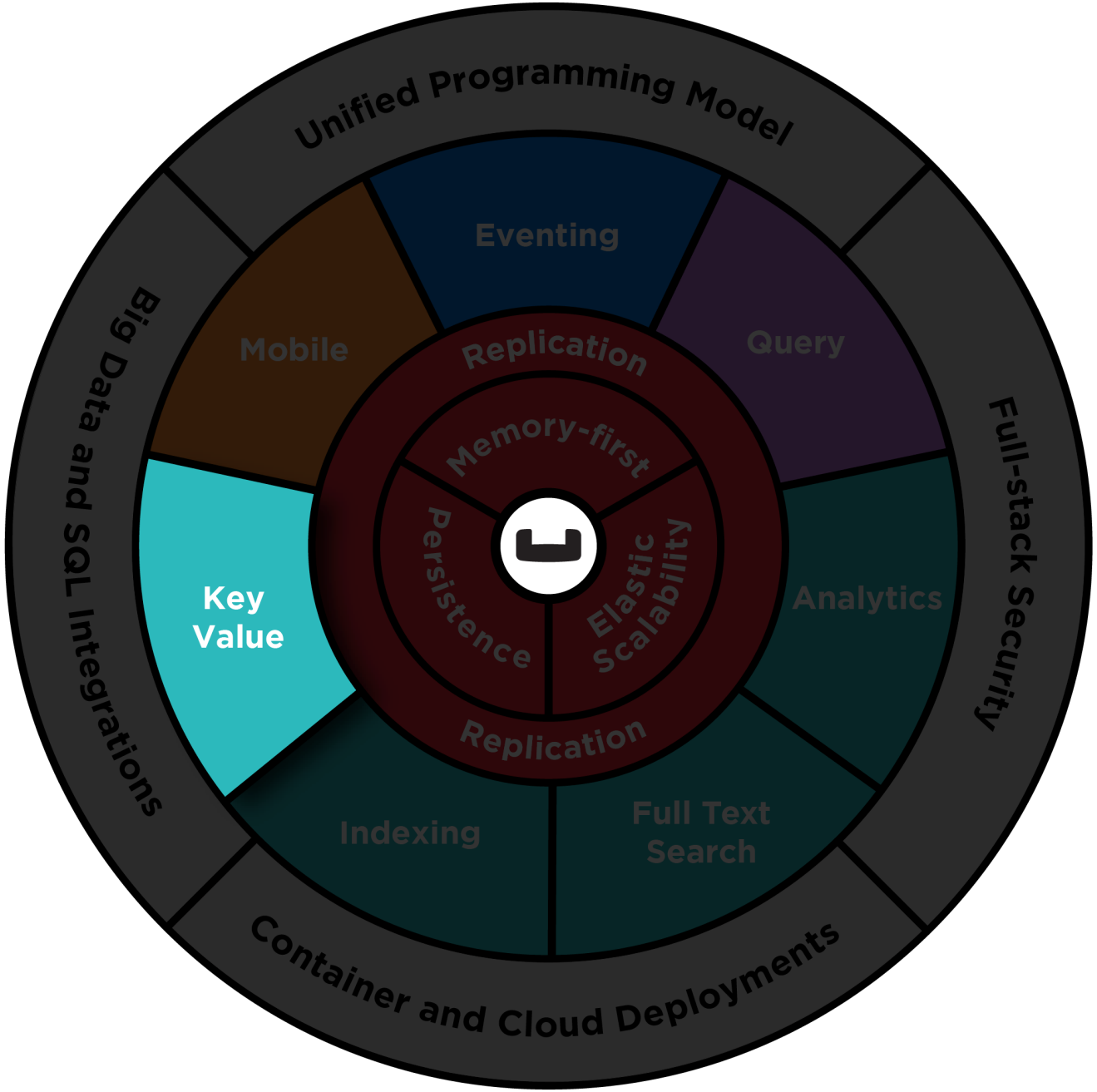
Cluster Manager - Rest API



https://developer.couchbase.com/documentation/server/current/rest-api/rest-endpoints-all.html

2 Couchbase Services

KEY VALUE

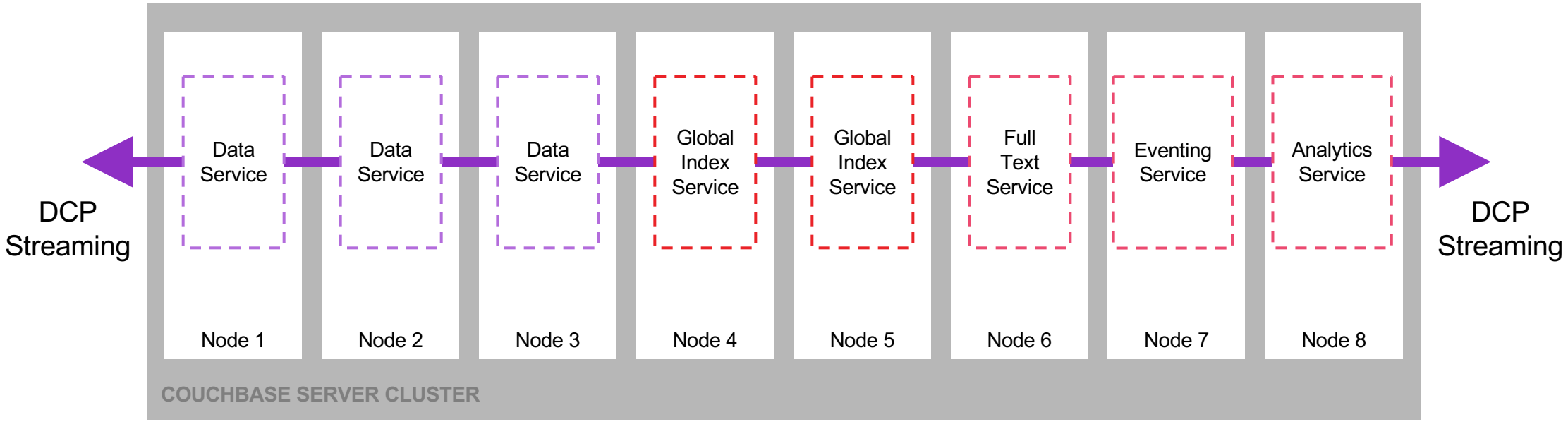

Unified Programming Model
Full-stack Security
Container and Cloud Deployments
Big Data and SQL Integrations
Eventing
Query
Analytics
Full Text Search
Indexing
Key Value
Mobile
Replication
Memory-first
Persistence
Elastic Scalability
Replication

Key Value Service

- Core Memory-Centric data store
  - Tunable Distributed Cache
  - Distributed Persistent storage
  - Intra-cluster replication
  - Database Change Protocol
- Key Value API
  - Full and sub-document access
  - Binary data types
  - Non-document data structures
- Intercluster Replication (XDCR)
- Local Map/Reduce Views **(DON'T USE!)**
  - Local Indexes
  - JavaScript Engine
  - Near-team aggregates

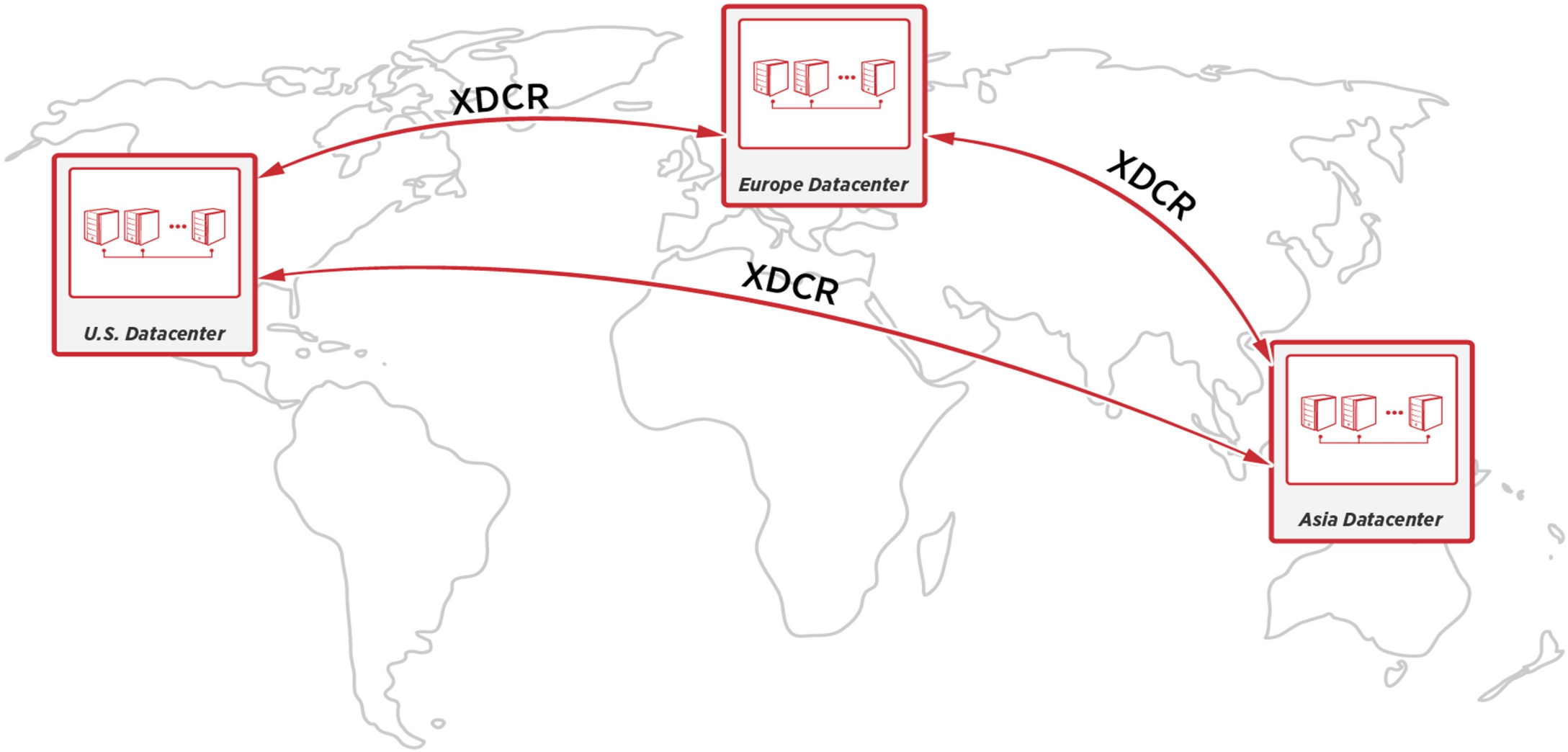
Memory-First Architecture

Data movement free from disk bottlenecks

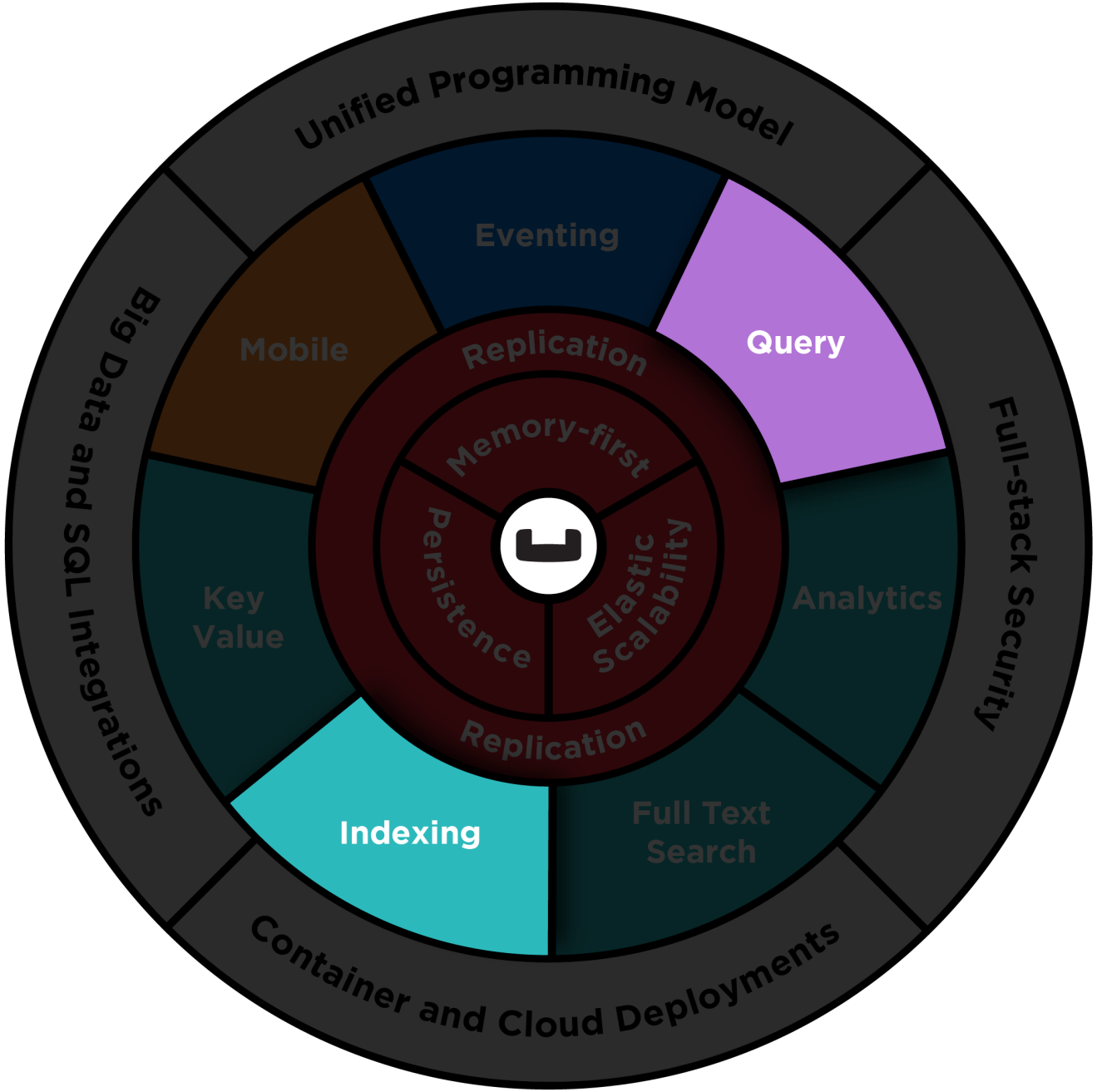


- In-memory streaming of updates to all components
- In-memory (cached) access to data and indexes
- Memory-only data buckets
- Memory-only indexes

Data Service - XDCR



QUERY AND INDEXING



Motivation: SQL for JSON

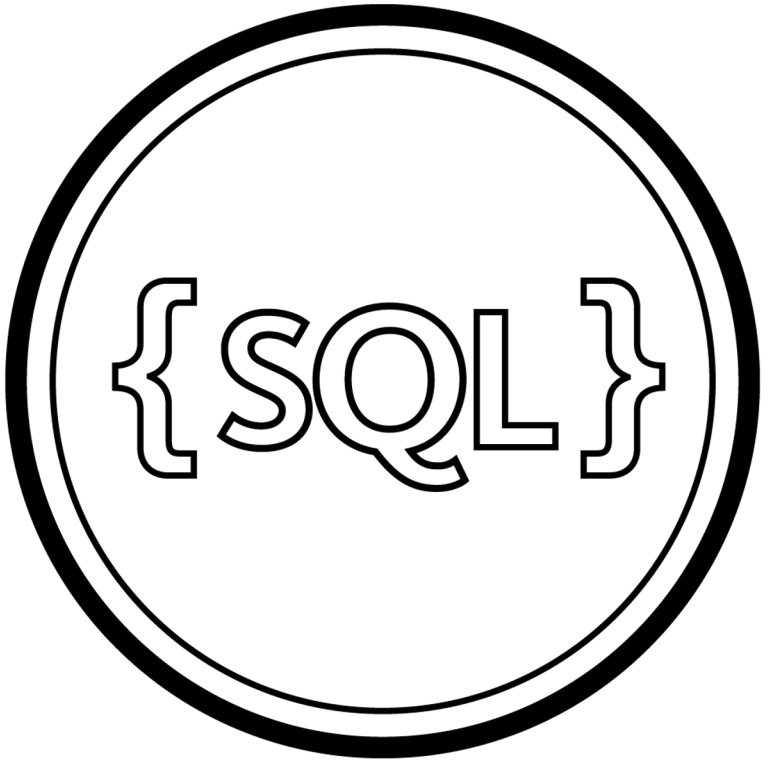
Give **developers** and **enterprises** an **expressive**, **powerful**, and **complete** language for **querying**, **transforming**, and **manipulating JSON** data.

Query and Index Services

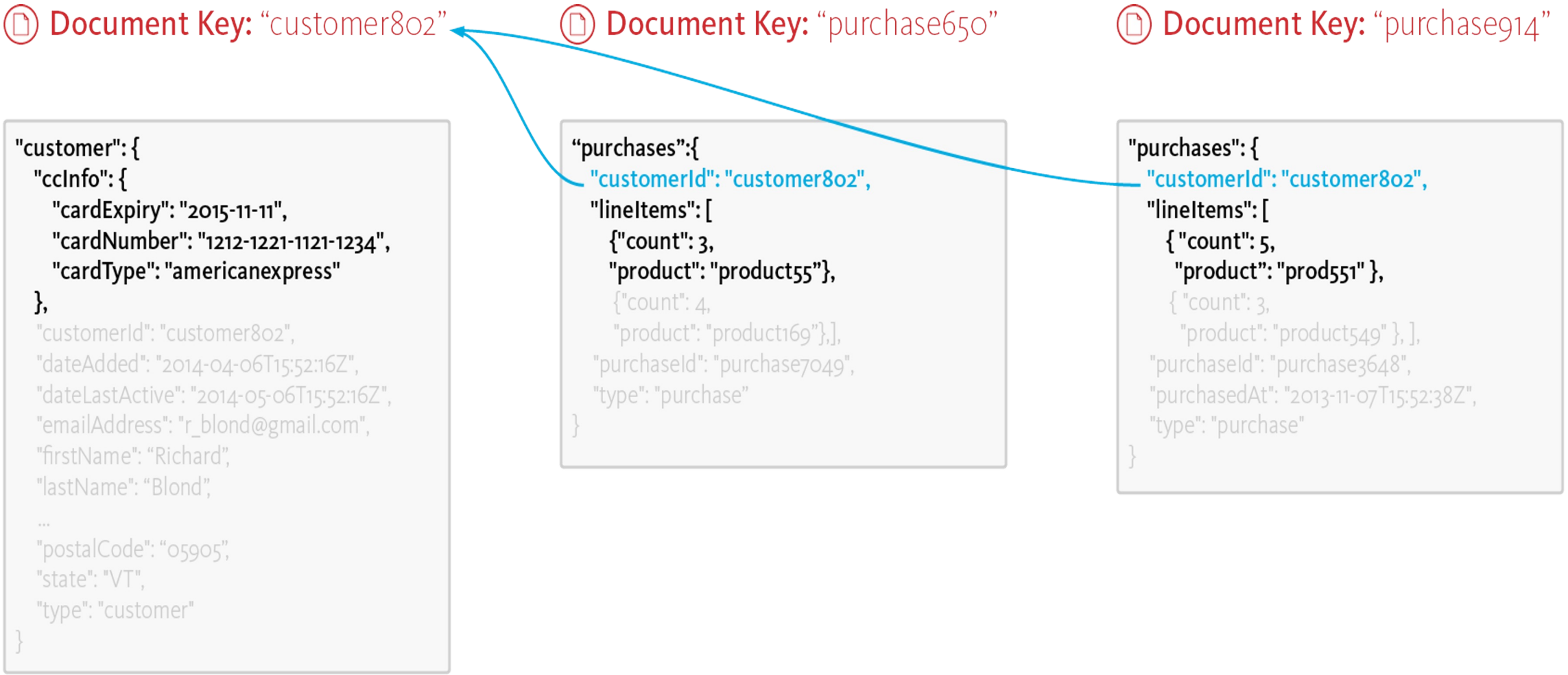
Fast Operational SQL Query and Global Indexing for JSON

- Highly-parallelized Query Engine
  - Multi-stage with in-stage multi-threading
  - Stateless / Horizontally scalable
- Memory Optimized Live Indexes
  - Read Your Own Writes
  - Tunable Consistency
  - Memory-to-Memory DCP projection
  - Extensive N1QL Syntax
- Capability for Unique Object-Centric Data

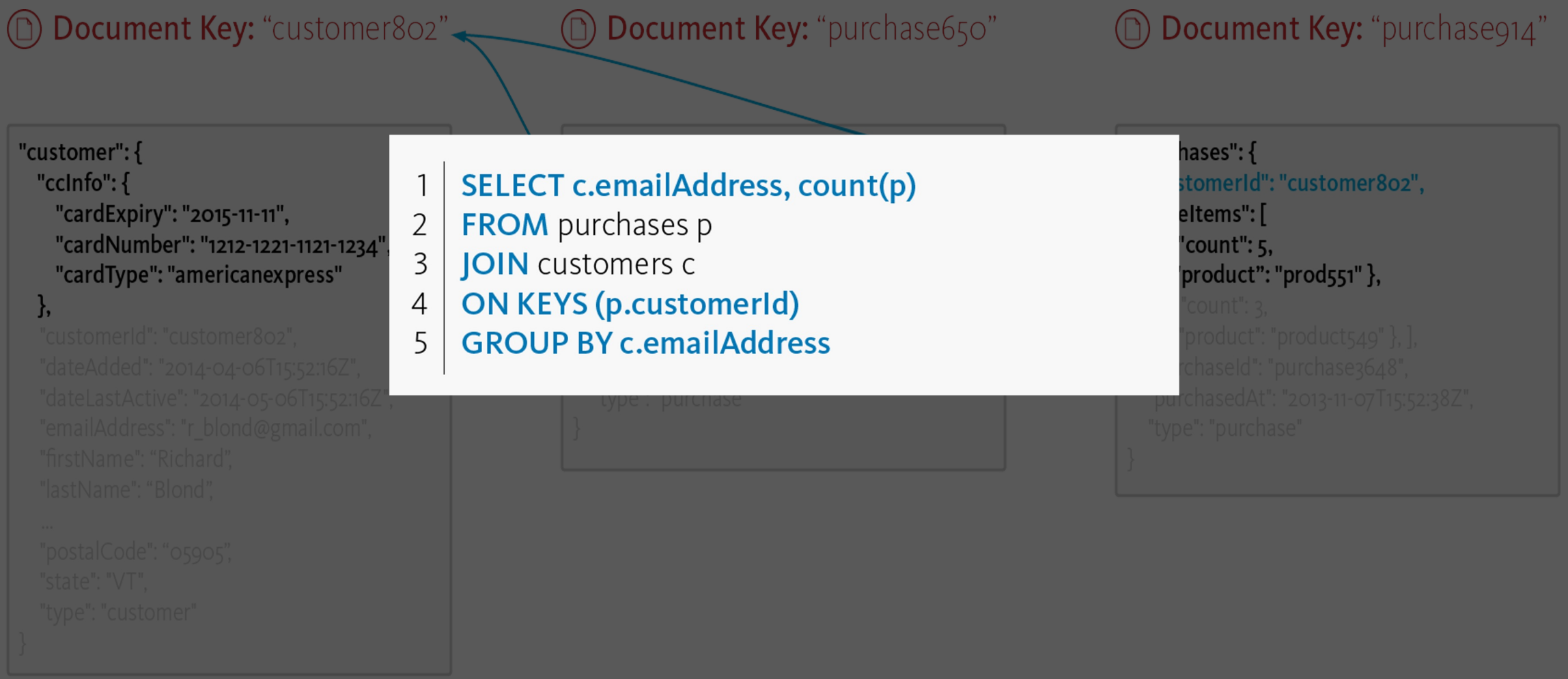
N1QL = JSON + SQL



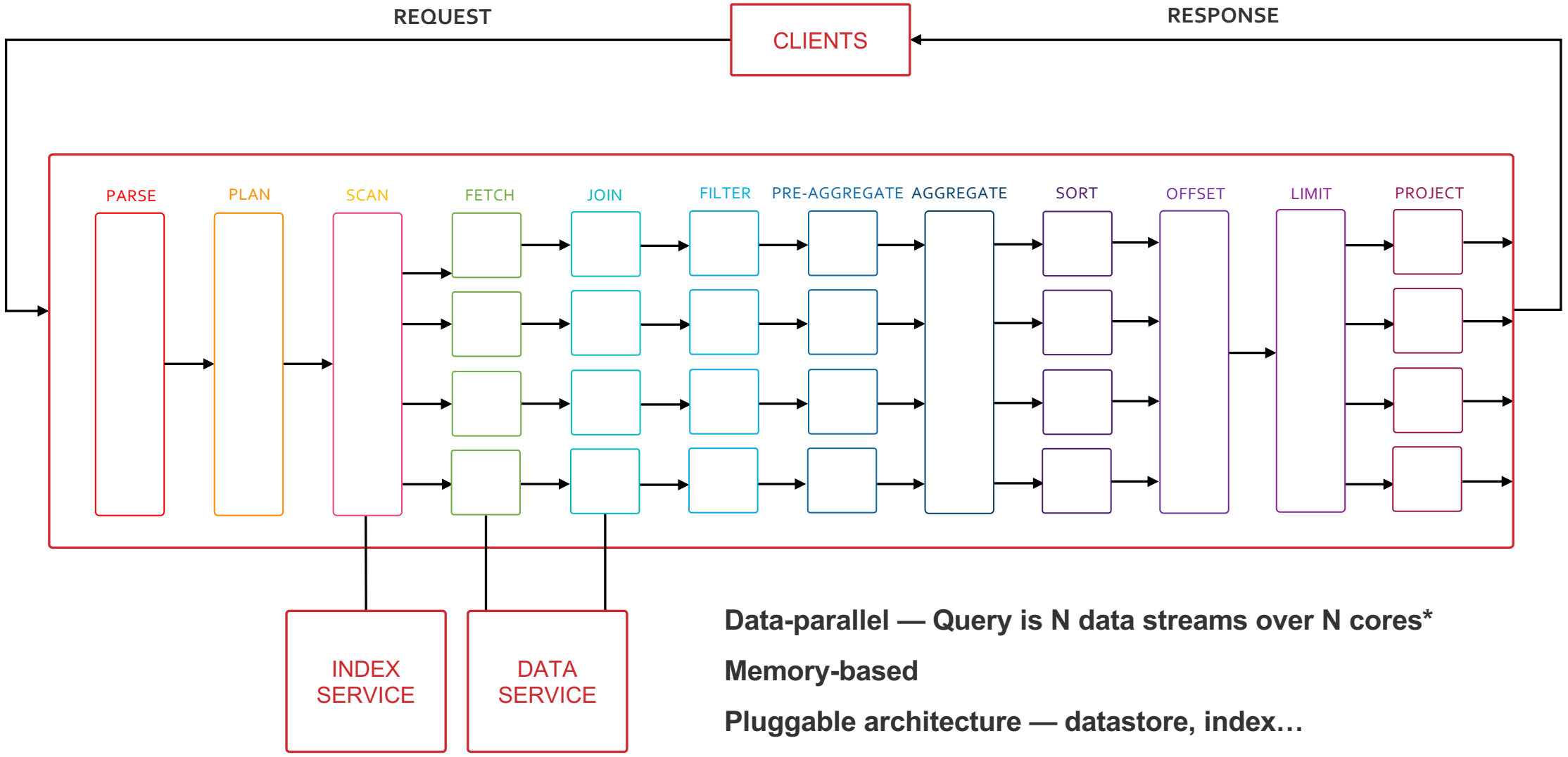
SQL++ is SQL for JSON



N1QL is SQL for JSON



Full Query Pipeline



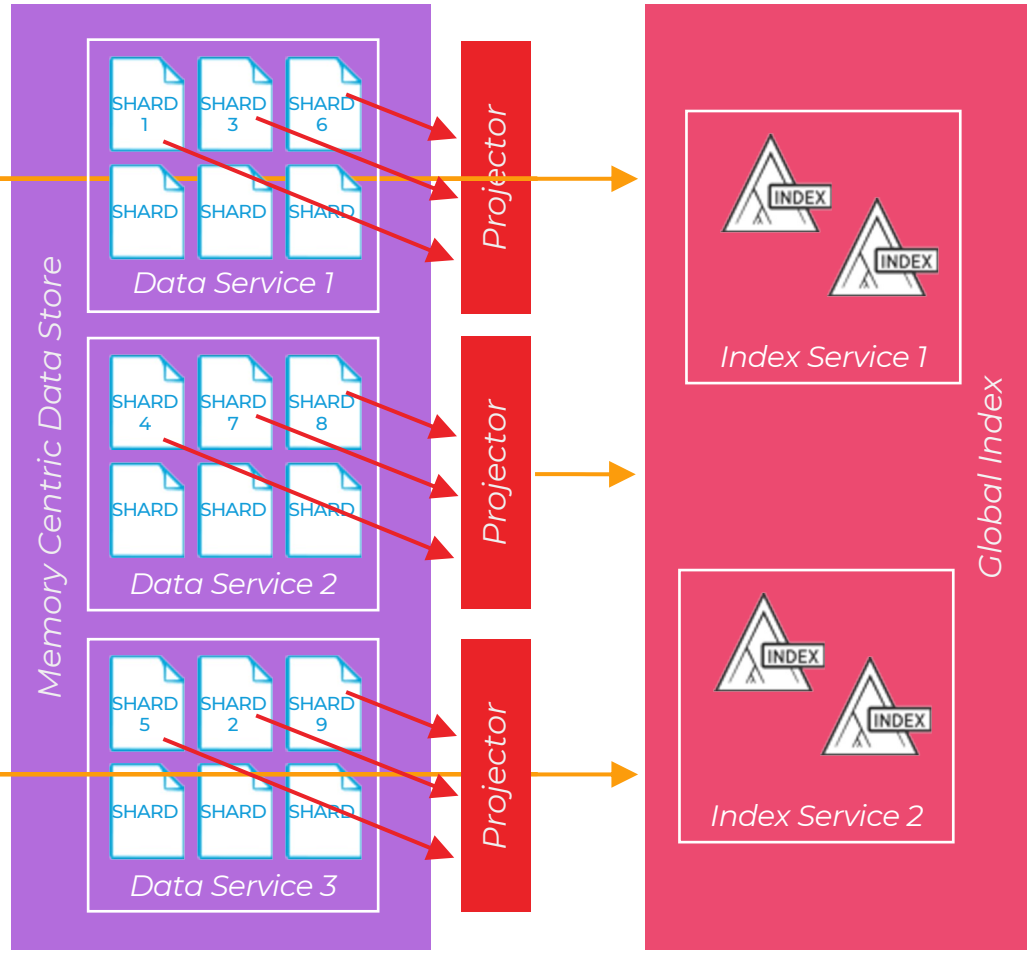
Data-parallel — Query is N data streams over N cores*

Memory-based

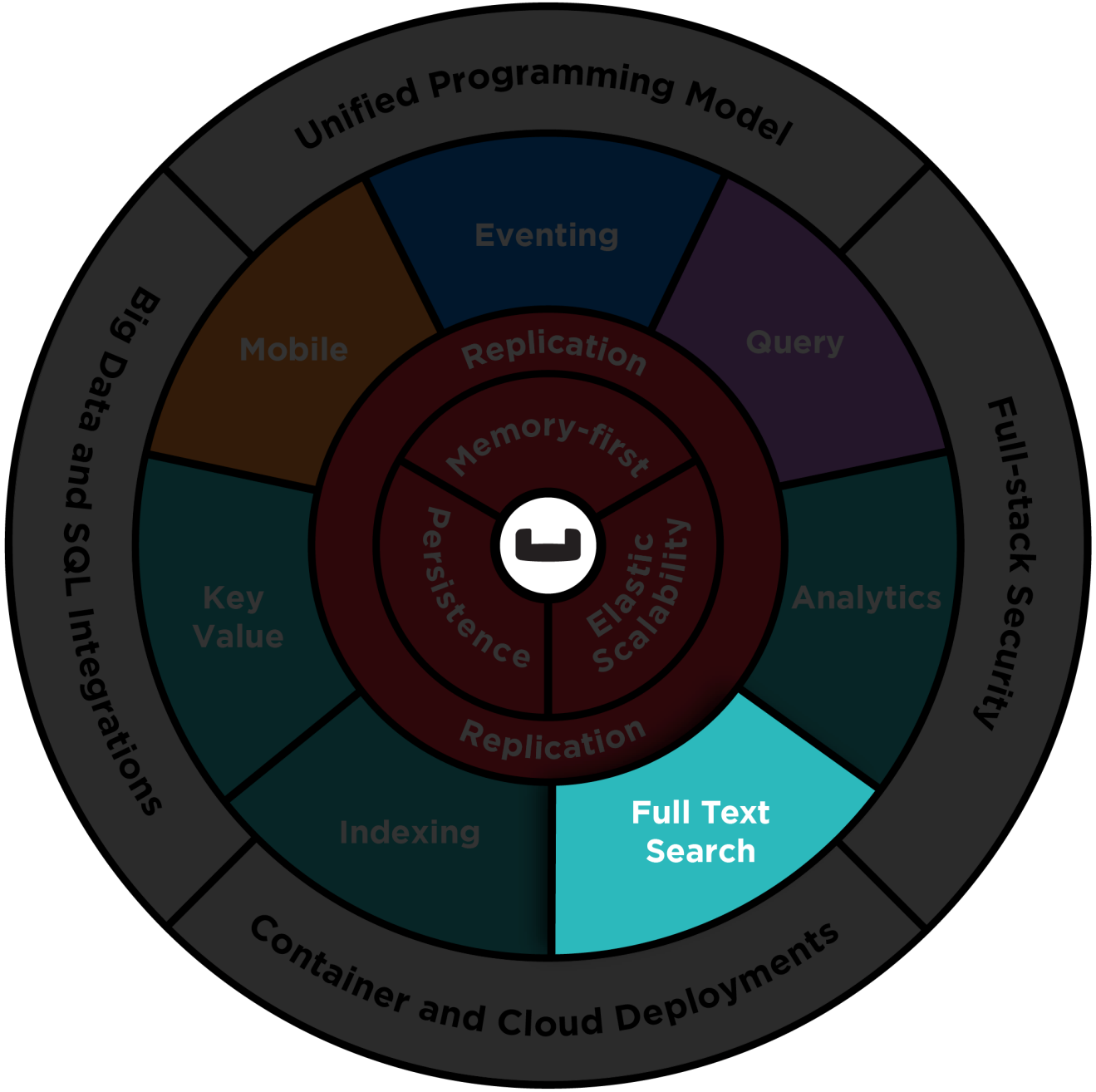
Pluggable architecture — datastore, index…

Global Indexes

- N1QL CREATE INDEX paradigm
- Indexing and scanning work is consolidated
- Indexer can process embedded functions and perform nested array traversal
- Each data service projects its own data
- Index replicas are supported
- Indexer can be pushed various scan types



FULL TEXT SEARCH



Full-Text Search Service

- Search / Query
  - Basic
  - Compound
  - Range
  - Special Purpose
  - Scoring
  - Geospatial
  - Custom Sort Ordering
- Indexing
  - Real time indexing
  - Default map and map by document type, or dynamic
  - Store fields, Term vectors
  - Analyzers: Tokenization, Token Filtering (stop word removal, stemming)
  - 20 supported languages

Index, Analyze and Query

1. **Index fields of a document**
   "...located in the heart of the new City Quay development. The hotel has views of ..."

2. **Analyze terms for index**
   located ... heart .. new City Quay development. .. hotel has views

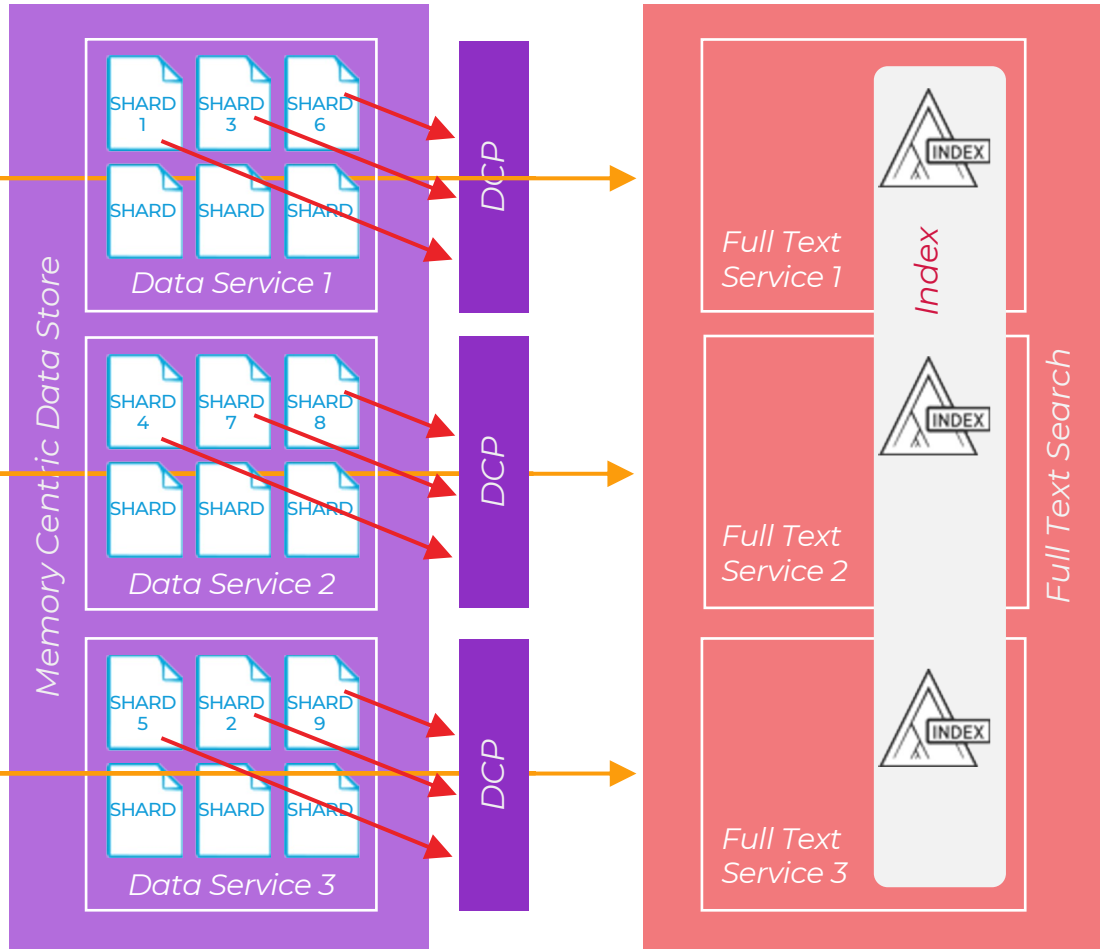
3. **Query the index**
   description: *location*

Return scored documents list

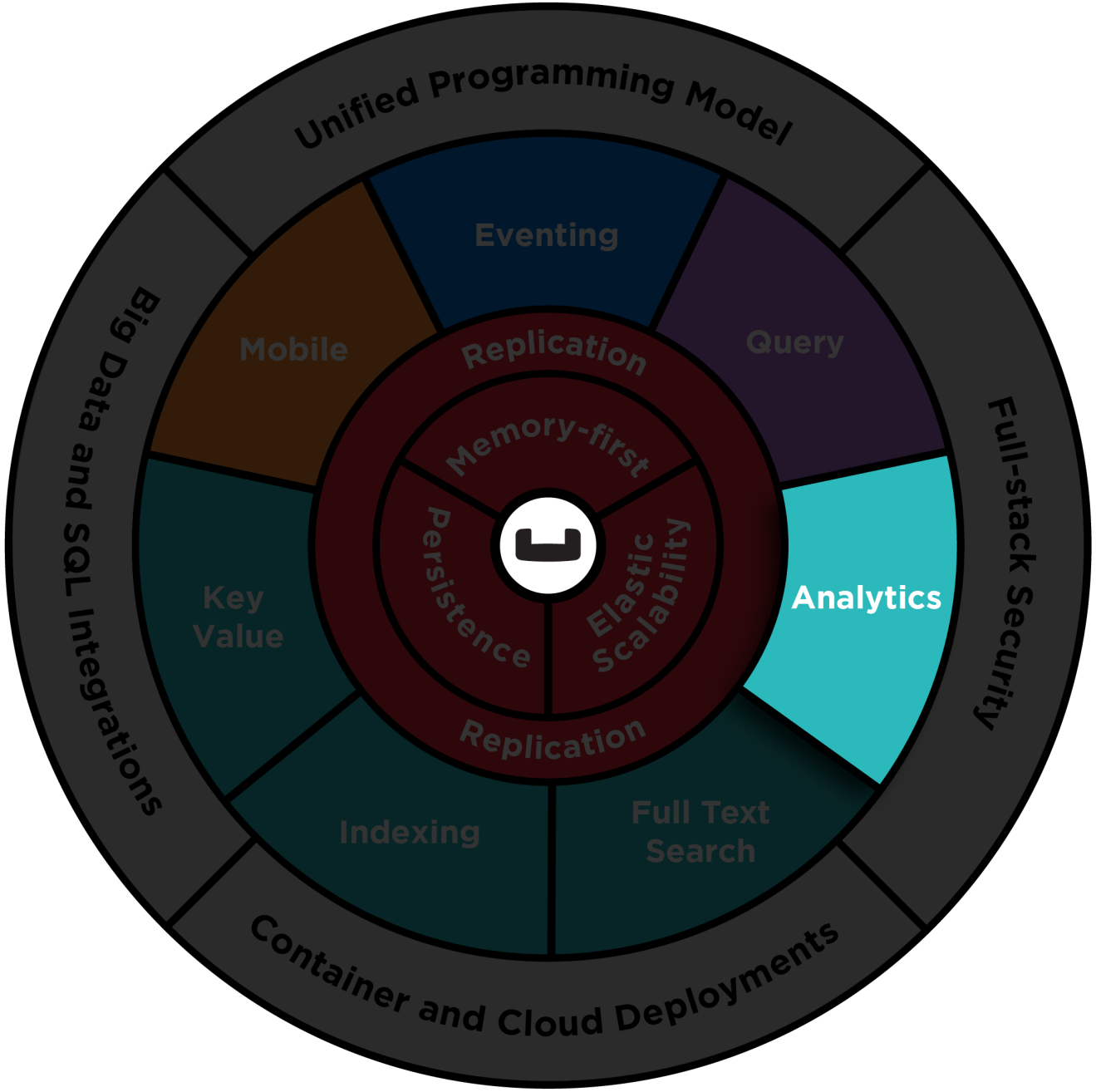
Scoring	Document ID	Description Matches
1.25	hotel_1234	best location
1.37	hotel_2345	loved hotel location
1.82	hotel_3456	location is awesome
1.88	hotel_4557	hard to locate

Full-Text Data Projection

- Each FTS node is a DCP stream subscriber
- Indexing work is distributed across FTS nodes
- ANY FTS service can receive FTS query, “scatters” to other nodes and “gathers” response
- Application sees a single logical index

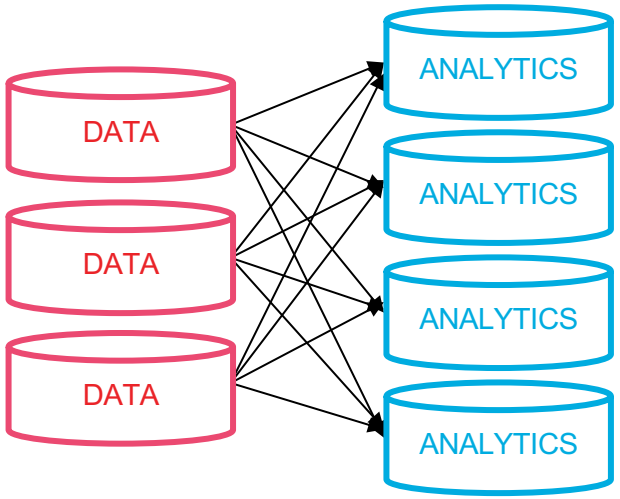


ANALYTICS


Unified Programming Model
Full-stack Security
Container and Cloud Deployments
Big Data and SQL Integrations
Eventing
Query
Mobile
Key Value
Indexing
Full Text Search
Analytics
Replication
Memory-first
Persistence
Elastic Scalability
Replication

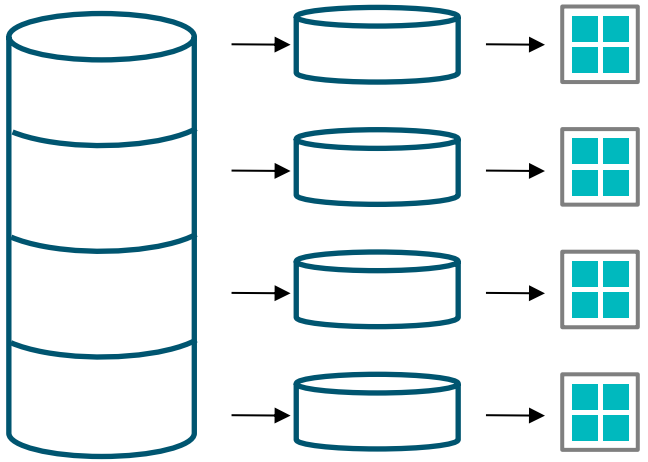
What is Couchbase Analytics?

Shadow Dataset of a Couchbase Data node

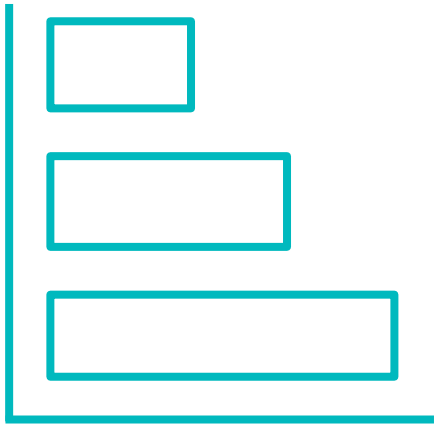


Fast Ingest

MPP architecture: parallelization among core and servers



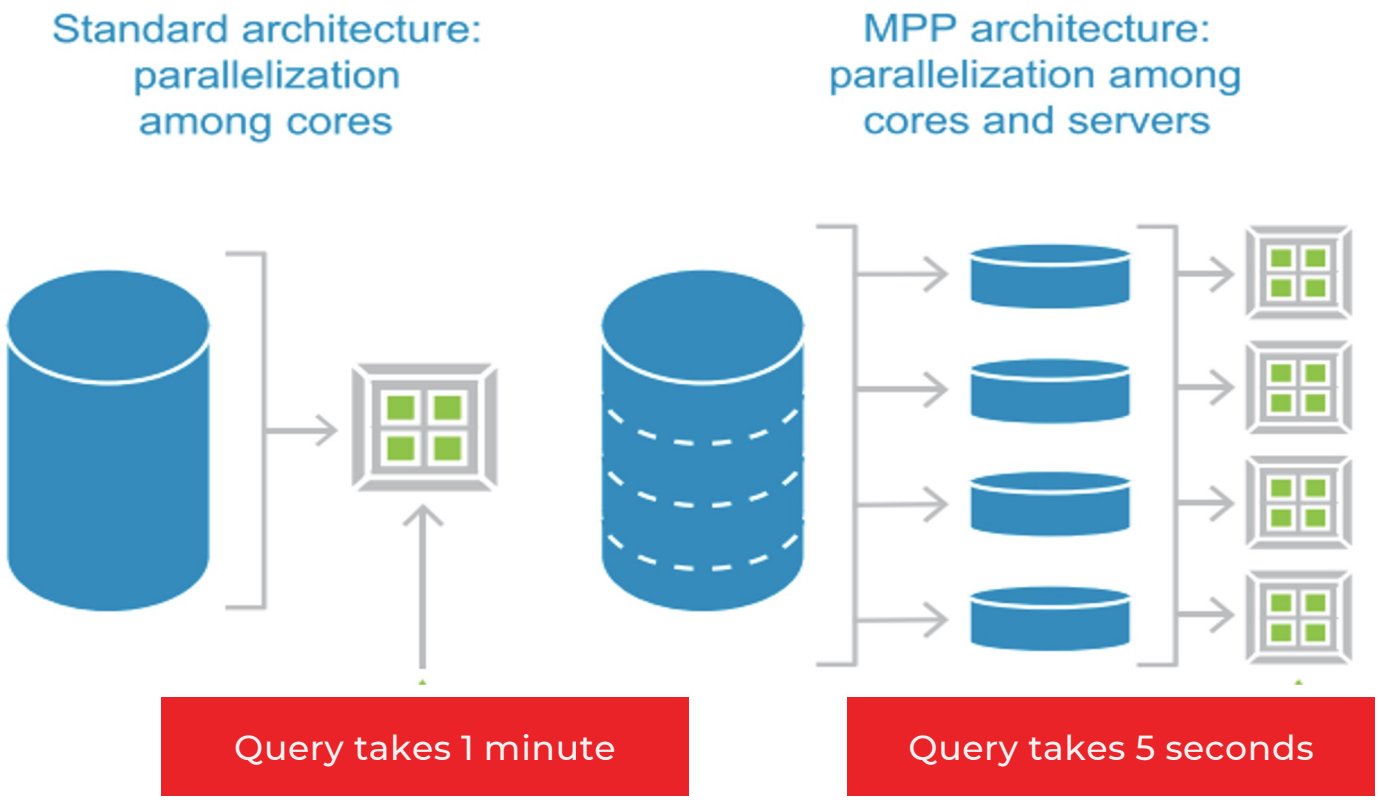
Complex Queries on large datasets



Real-time Insights for Business Teams

Distributed Complex Query Processing

- Parallel Query Processing
  - Distributed Massively Parallel Query Processor (MPP) quickly executes complex queries on larger datasets
  - Comprehensive SQL-like query language (SQL++)

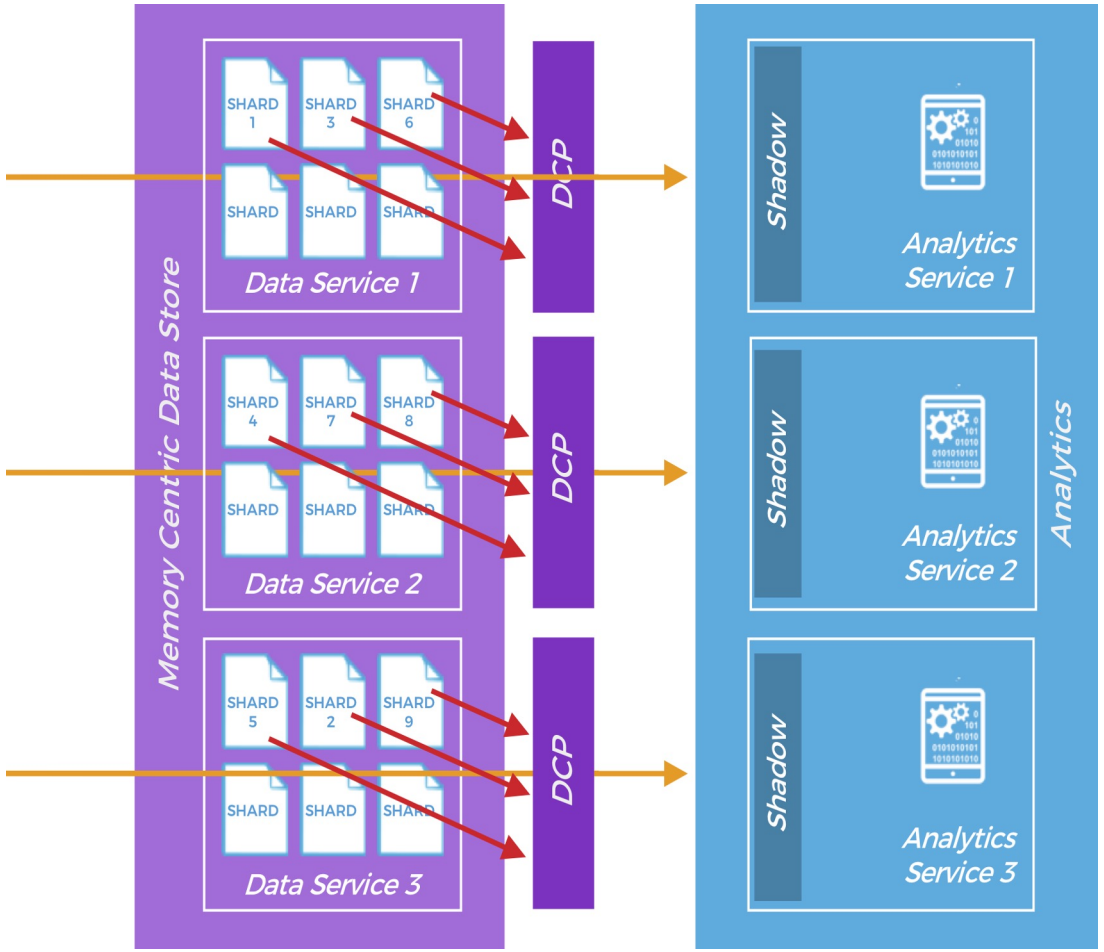


Analytics Service

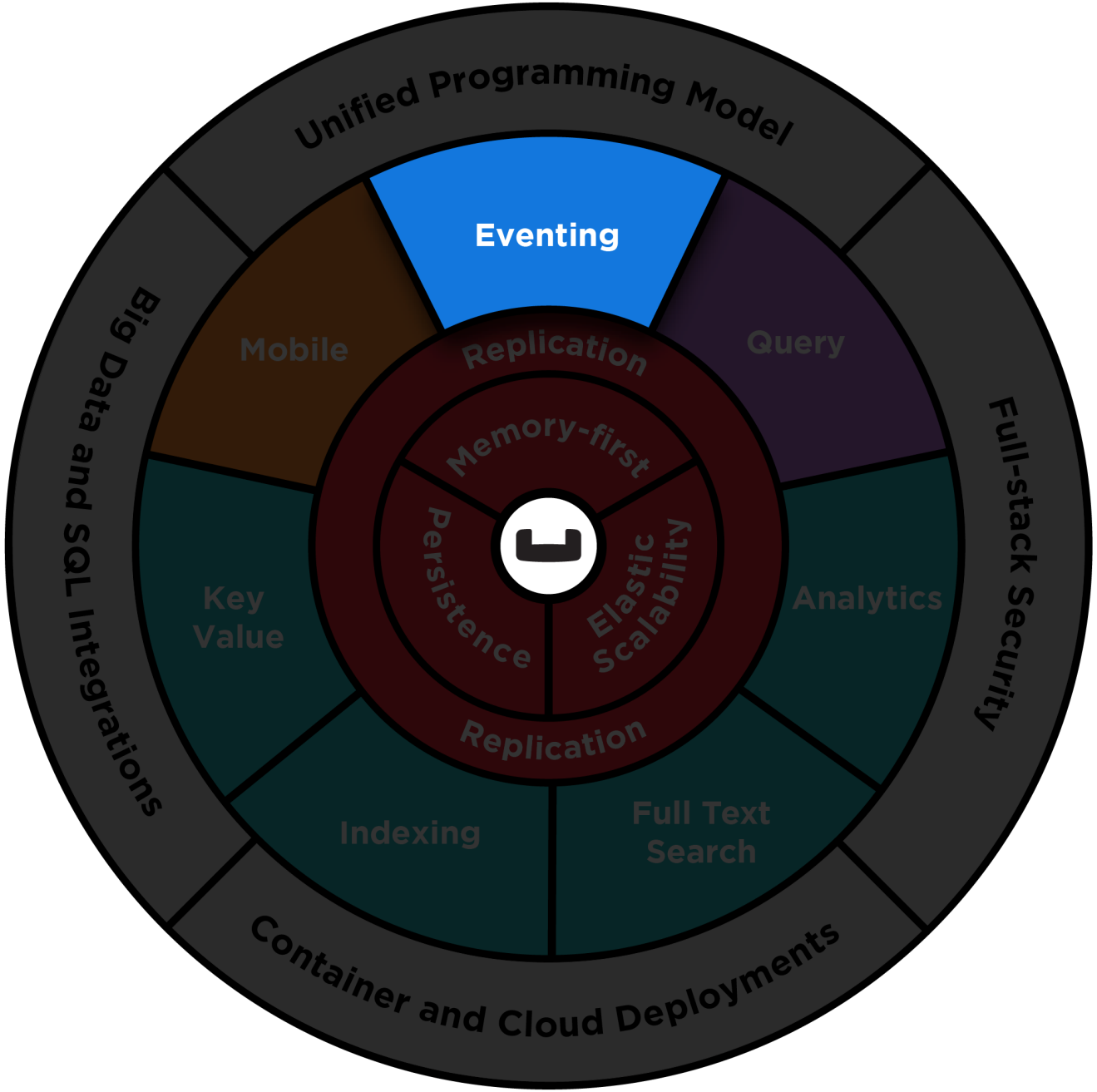
- Common Programming and Data Model
  - Completed, declarative query language, well-suited for modern NoSQL data models
- Ad-Hoc Queries
  - Ask anything, no explicit index creation required
- Fast, Complex Queries
  - Large join, set, aggregation, grouping operations
- Workload Isolation
  - Runs on specialized nodes, no performance impact on the core operational cluster
- Data Freshness
  - Fast memory-to-memory synchronization with the core data cluster
- Scalable & Secure
  - The same horizontally scalable, secure cluster infrastructure

Bucket Shadowing

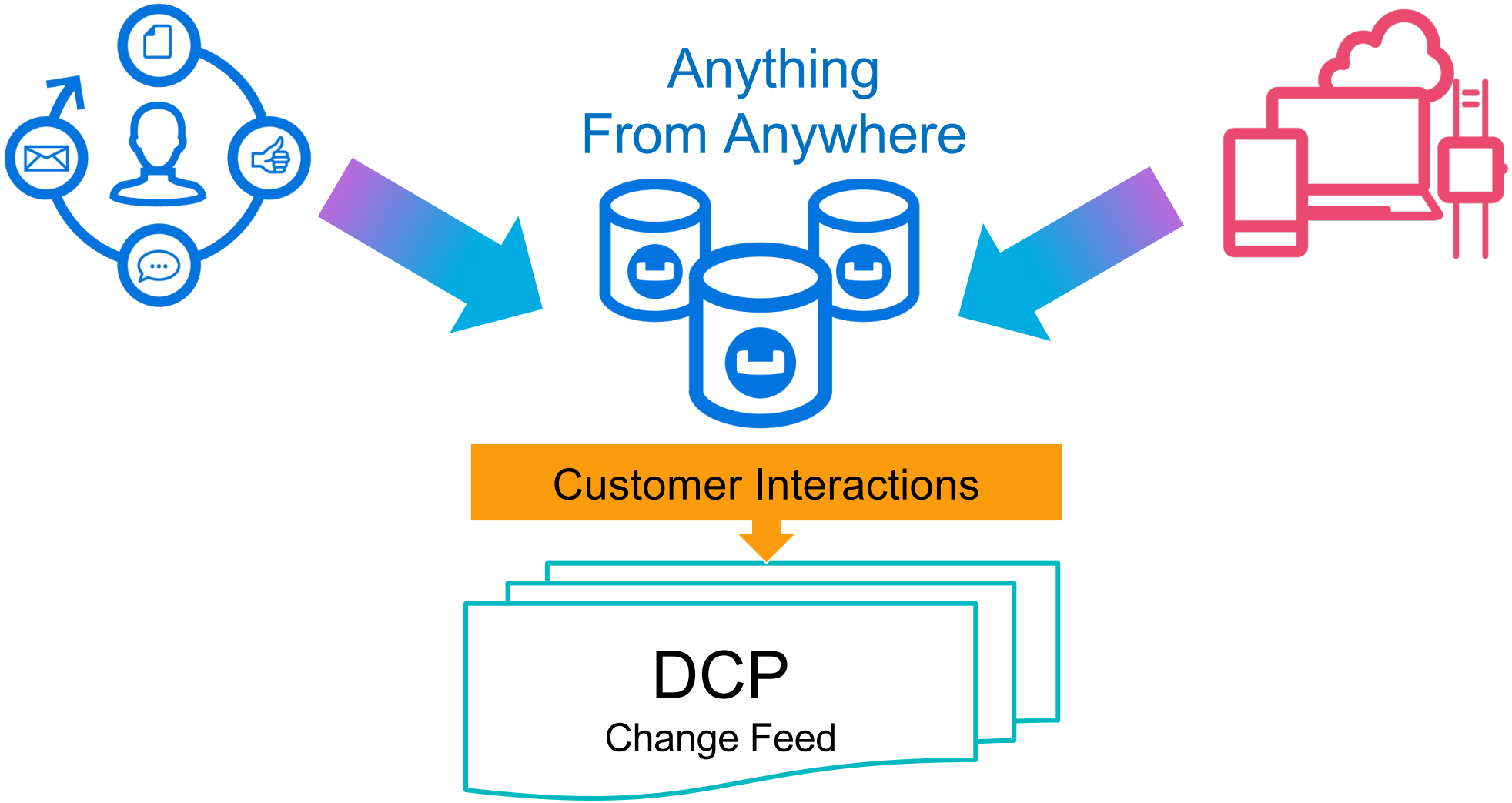
- Data Change Protocol (DCP) automatically synchronizes data to shadow sets
- Shadow is a Query-specific distribution and organization of data, aggregation, index and query processing
- Longer term data can be persisted even when operational data is expired/deleted
- Operational data store experiences minimal impact from even the most complex and long-running queries



EVENTING SERVICE



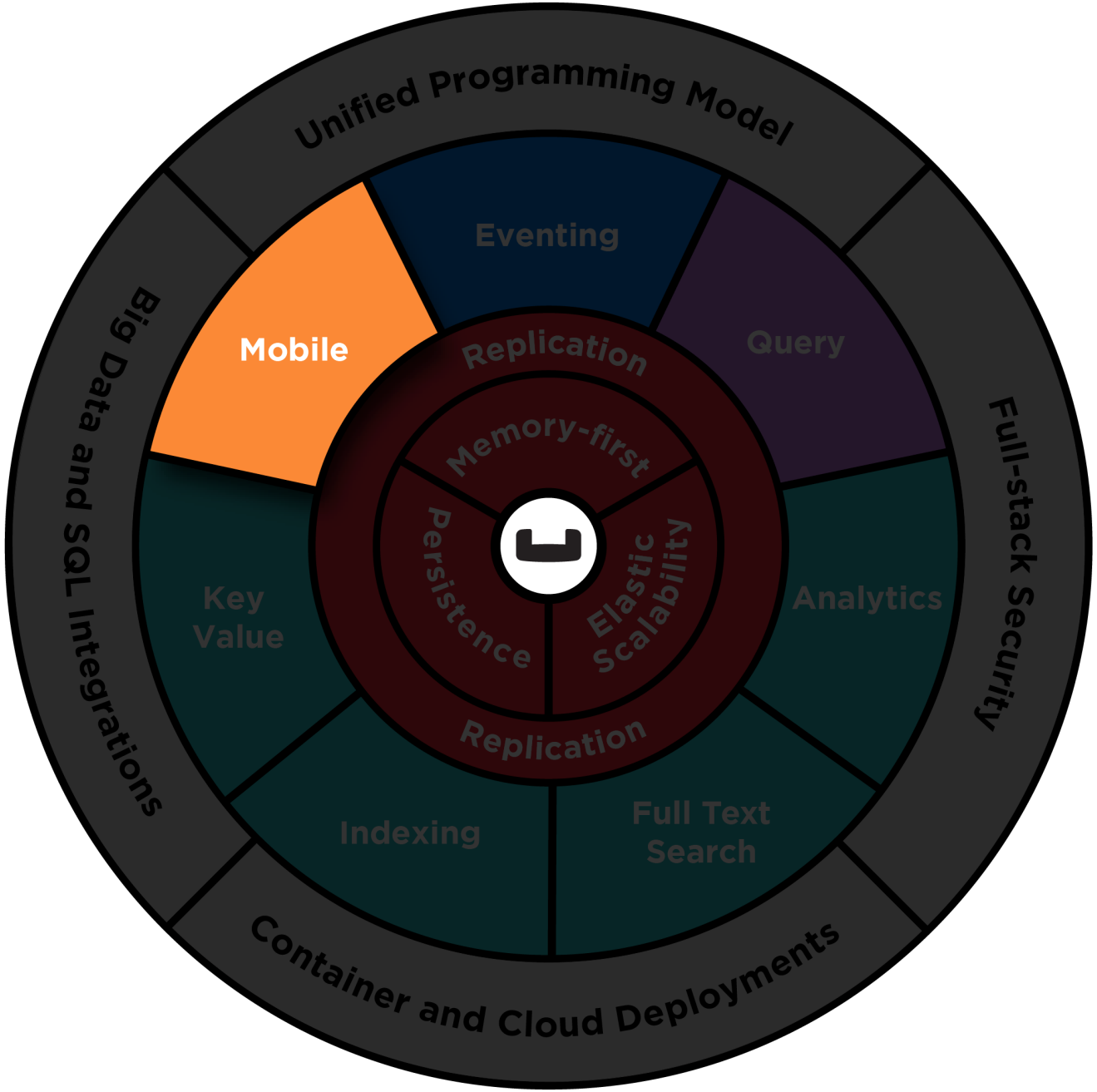
What is Eventing Service?



Eventing Service

- MDS enabled service
- Event-Condition-Action model for data change
- Stateless compute for low-latency workloads
- Highly performant and fault-tolerant service at scale
- Handlers for:
  - OnUpdate (for insert/update)
  - OnDelete
- Functions enable:
  - Notification before expiry
  - Threshold-based monitoring and alerting
  - Propagation of changes to other systems
  - Enrichment of content in near real-time
  - Triggering of routines on given documents at specified intervals
  - Cascade of deletes
  - And More!

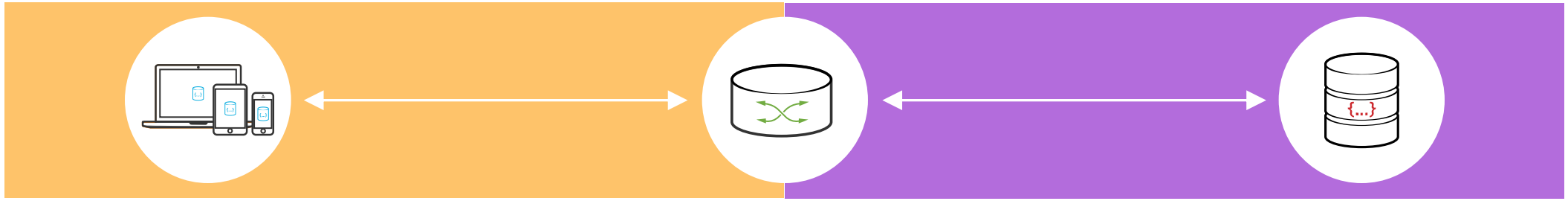
MOBILE


Unified Programming Model
Full-stack Security
Container and Cloud Deployments
Big Data and SQL Integrations
Mobile
Eventing
Query
Analytics
Full Text Search
Indexing
Key Value
Replication
Memory-first
Persistence
Elastic Scalability
Replication

Mobile with “Offline First” Sync

- Offline First
  - On-device in-app document store
  - Automatic conflict resolution
  - Data-driven alerts
- Replication
  - Low-latency sync with WebSocket
  - Channel-centric access
  - Dynamically tunable
- N1QL
- Full-Text Search

Couchbase - The Data Platform for Mobile Engagement



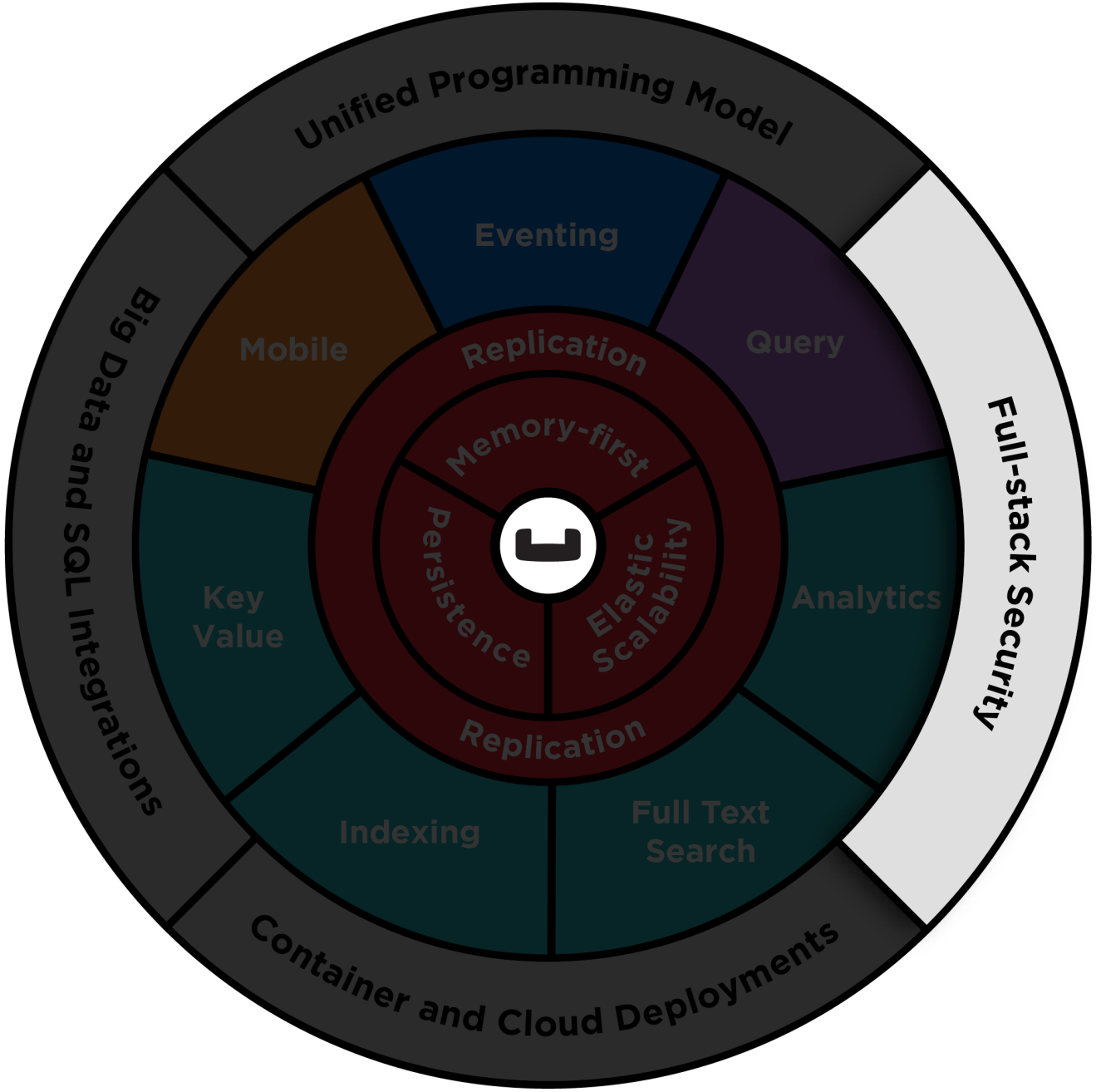
COUCHBASE LITE | **SYNC GATEWAY** | **COUCHBASE SERVER**

Client	Middle Tier	Storage
Lightweight embedded NoSQL database with full CRUD, query and FTS functionality.	Secure web gateway with synchronization, data access, and data integration APIs for accessing, integrating, and synchronizing data over the web.	Highly scalable, highly available, high performance NoSQL database server.

Security

Built-in enterprise level security throughout the entire stack includes user authentication, user and role based data access control (RBAC), secure transport (TLS), and 256-bit AES full database encryption.

SECURITY

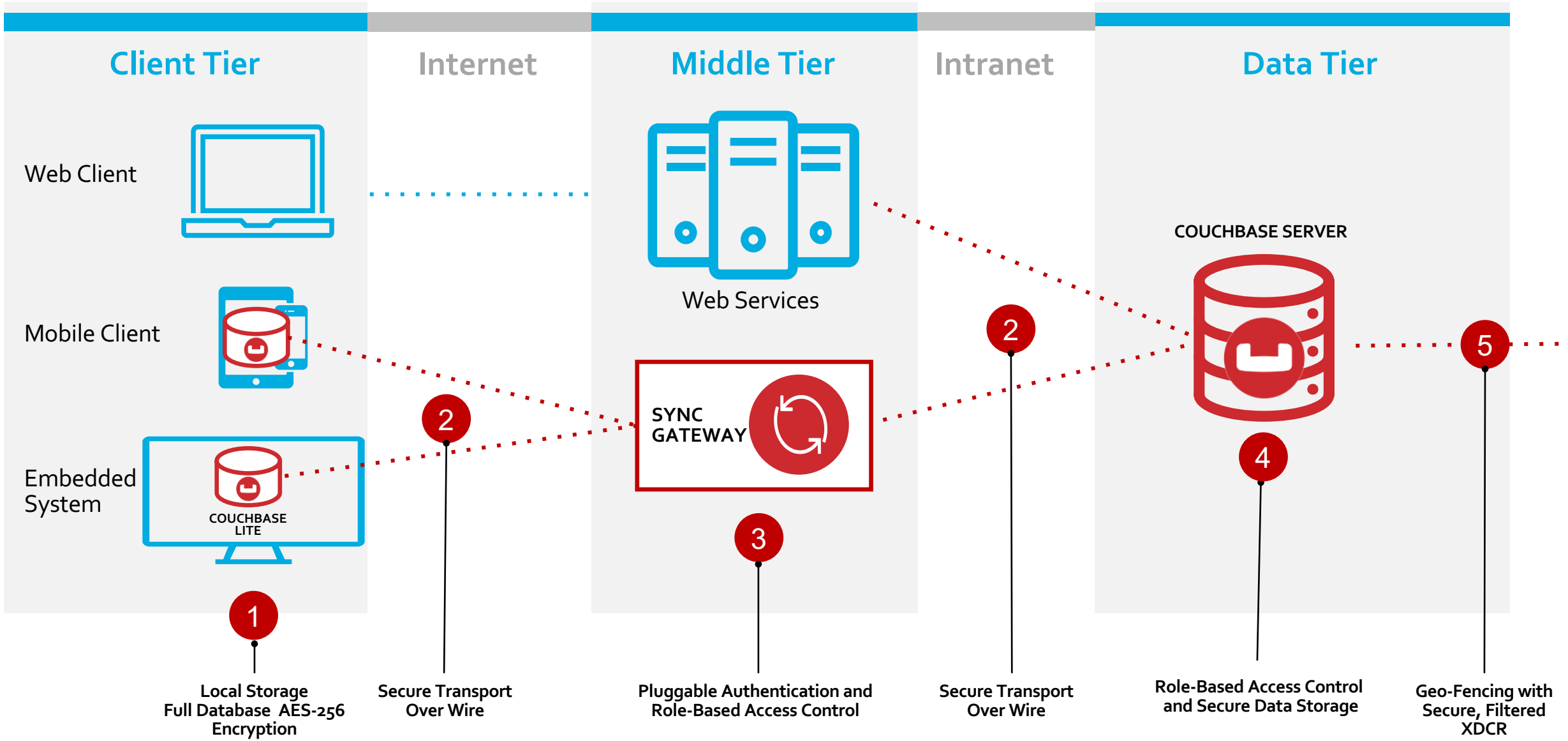

Unified Programming Model
Full-stack Security
Container and Cloud Deployments
Big Data and SQL Integrations
Eventing
Mobile
Query
Key Value
Analytics
Indexing
Full Text Search
Replication
Memory-first
Persistence
Elastic Scalability
Replication

Security Pillars in Couchbase

Authentication	Authorization	Crypto	Auditing	Operations
App/Data: SASL AuthN Admin: Local or LDAP PAM Authentication (4.6)	Local Admin User Local Read-Only User RBAC for Admins RBAC for Applications (5.0)	TLS admin access TLS client-server access Secure XDCR X.509 certificates for TLS Data-at-rest Encryption* Field-level Encryption (5.5)	Admin auditing	Security management via UI/CLI/REST

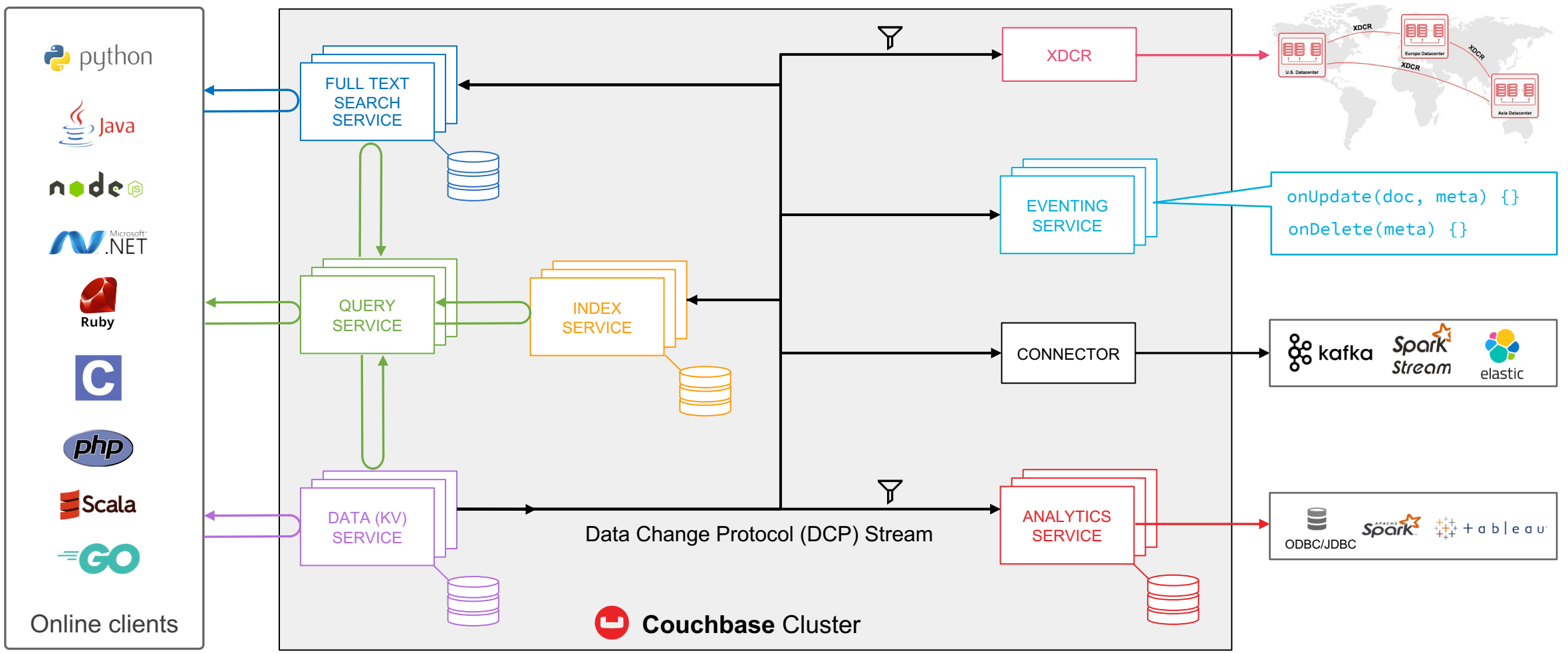
* Via third-party partners

Couchbase addresses Security concerns for the full stack



3 Core Engine Design

Couchbase Server Architecture



Operational — *Analytic / Ad hoc*

Unified Programming Interface

Application embeds the Couchbase SDK that provides a unified API to the Couchbase distributed services

COUCHBASE CLIENTS

GO C Java Microsoft .NET node.js php python Scala Ruby

Key Value

```
INSERT("key1",doc)

GET("key1)
```

N1QL query

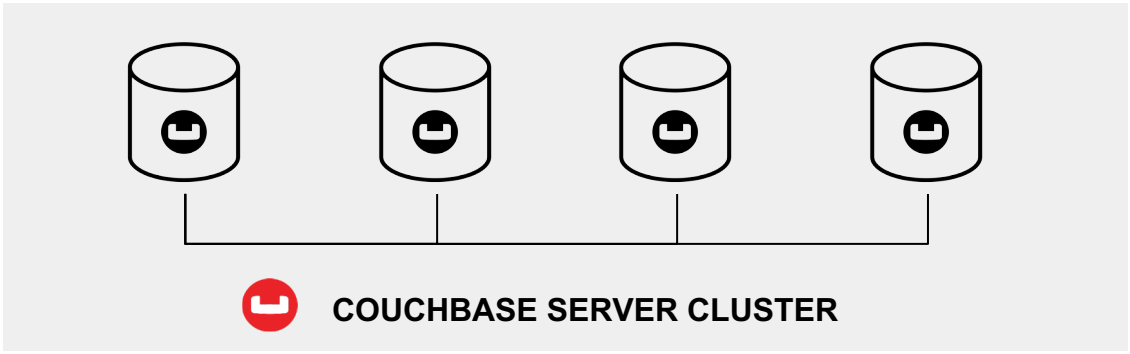
```
SELECT a.country
FROM airline a
WHERE a.name = "Excel
Airways";
```

Full Text Search

```
{
  "fields": ["*"],
  "query": {
    "query": "+nice +view"
   }
}
```

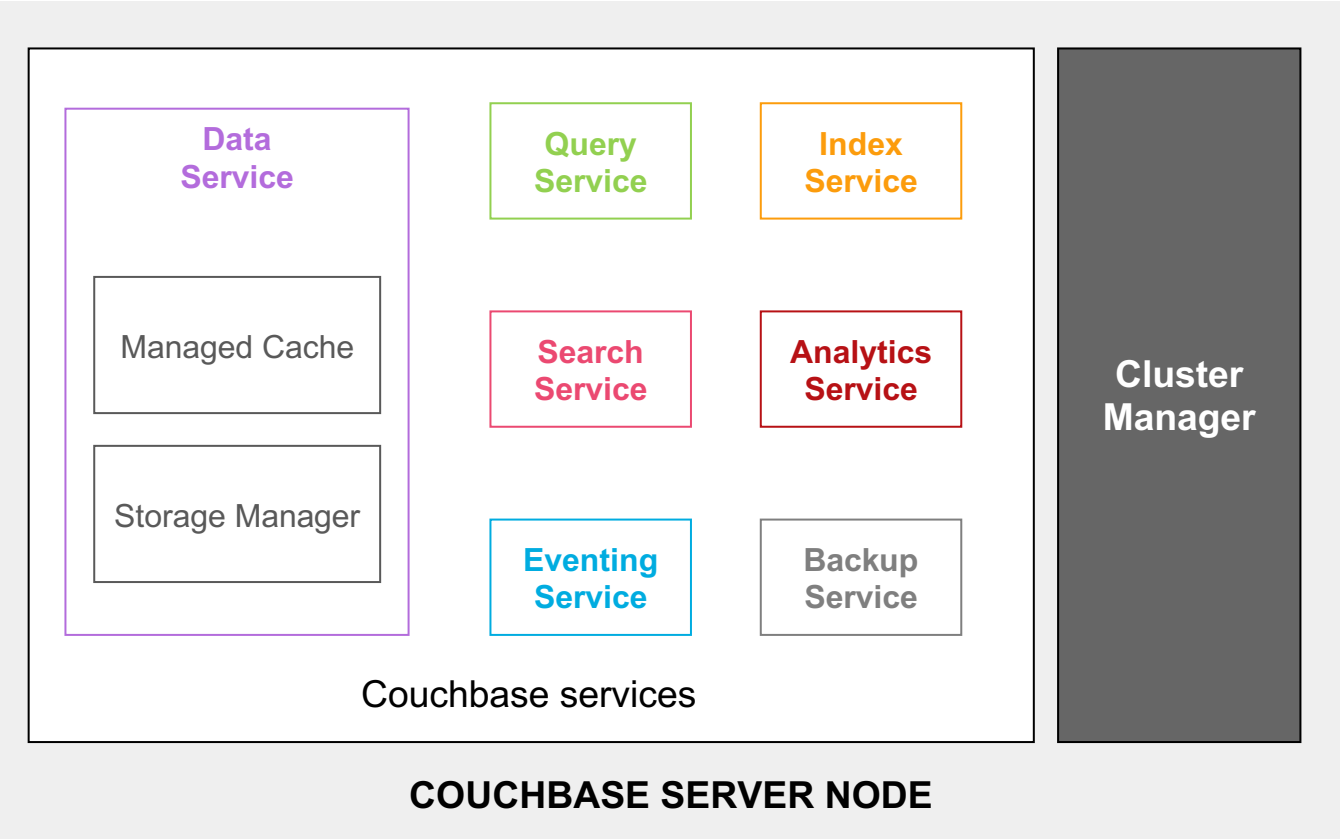
Transaction

```
START TRANSACTION;
UPDATE cust SET b=b-1000
WHERE user="User1";
UPDATE cust SET b=b+1000
WHERE user="User2";
COMMIT ;
```



Single Node Deployment

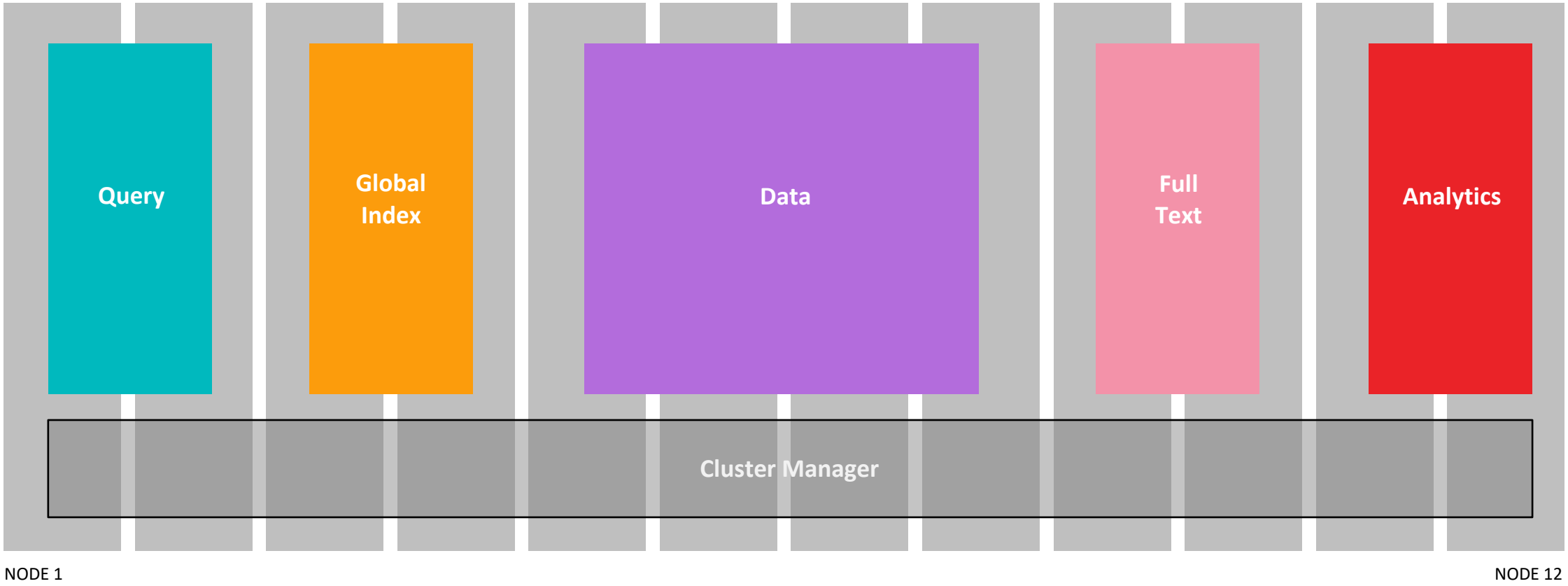
Couchbase Server consists of a single package that is installed on all nodes



- **Data Service** – Key-Value data access. The most fundamental of all Couchbase services, providing access to data in memory and on disk.
- **Query Service** – Parse and execute N1QL queries
- **Index Service** – Creates indexes, for use by the Query service.
- **Search Service** – Supports Full Text Search indexes and queries.
- **Analytics Service** – MPP query engine to perform complex joins, groupings, aggregations and count.
- **Eventing Service** – Operates on changes to data in real time
- **Backup Service** – Schedules full and incremental data backups
- **Cluster Manager** – configuration, heartbeat, statistics, RESTful Management interface

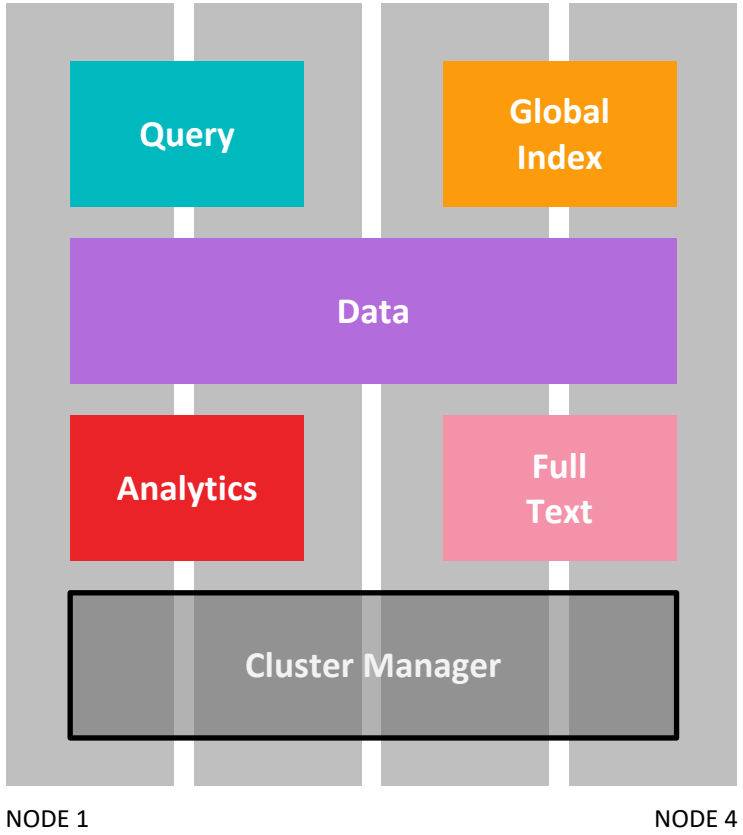
Multidimensional Scaling

Sample Production Deployment

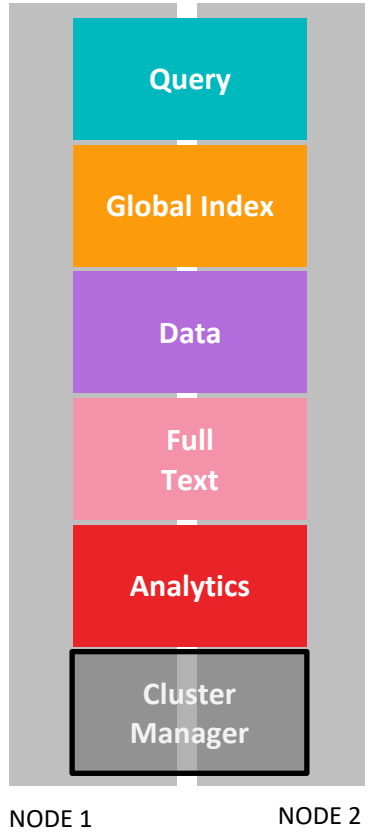


Multidimensional Scaling

Example Q/A

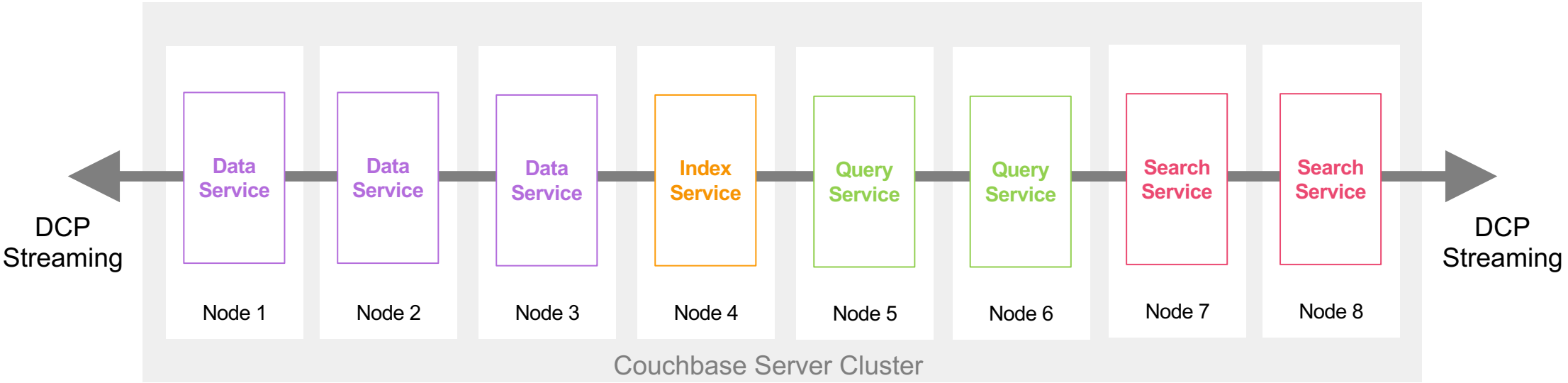


Example Development



Memory-first Asynchronous architecture

Data movement free from disk bottlenecks

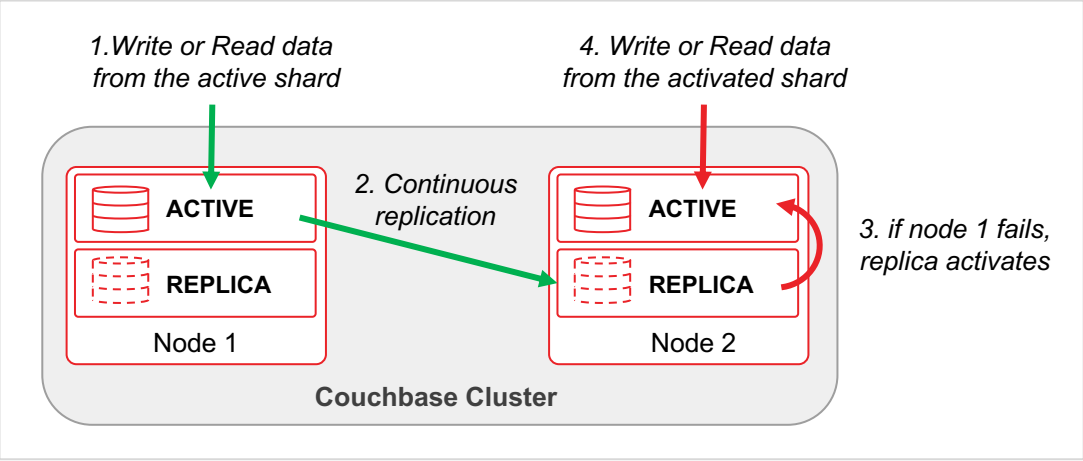


- In-memory streaming of updates to all components
- In-memory cache
- Memory-only data buckets
- Memory-only indexes

High Availability based on Replication

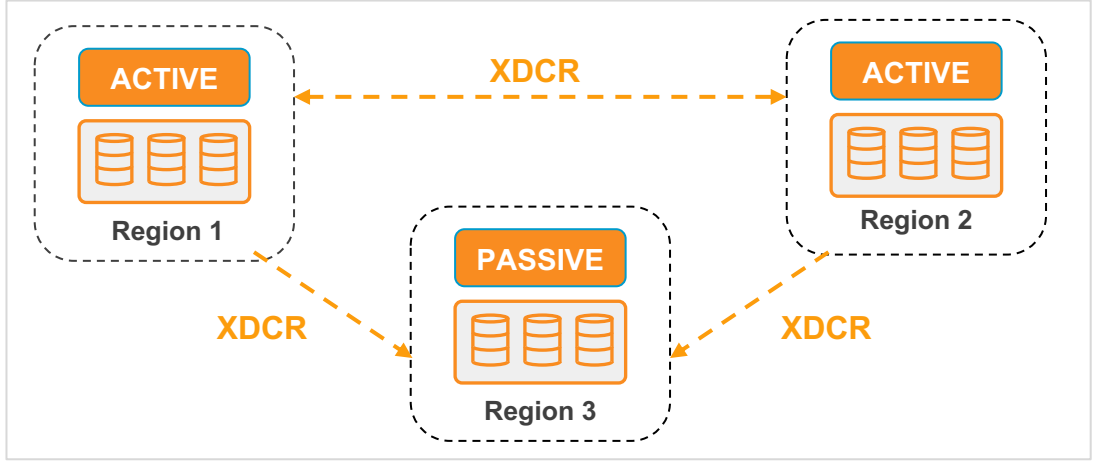
Couchbase Server ensures the availability of data across the nodes of a cluster and between clusters

Intra Cluster Replication



- Auto-sharding provides even distribution of data
- Data updates on the active shard are continuously replicated to their replica shards
- A replica is promoted in case of a node failure

Cross Data Center Replication



- XDCR allows data to be replicated across clusters located in different data centers
- Replication can be uni or bi-directional
- Replication can be fine-grained and filtered

Container and cloud deployment

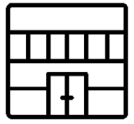
Multiple deployment options from DBaaS to Self-Managed

Database as a Service

- Fully-managed service
- Comprehensive monitoring and support

Self-Managed

- Maximize flexibility, customizability and performance
- Leverage DBA's and Infrastructure admins to manage

Hosted	Virtual Private Cloud	Public Clouds (On IaaS)	Your Clouds (On Prem)
			
• Launch in minutes • Single bill, no guesswork	• We manage in your VPC • Leverage your cloud account discounts	• Deploy via Kubernetes • Couchbase Cloud Native Database - full stack	• Maximum control

4 Use Cases

Who is using Couchbase ?

amadeus

GANNETT

50M Unique
monthly visitors
2.5B monthly page views
Replaced MongoDB

LinkedIn

> 450M members
16M entries every 5 min
10+ M queries/sec

PayPal

1 billion+ documents
10TB+ data
< 200 ms
response time

Retail

Walmart, cars.com, Best Buy, Fanatics, Staples, Office Depot OfficeMax, QVC, Tesco

Travel

Marriott, skyscanner, tripadvisor, Sabre, Orbitz, Ryanair, United

Gaming

EA, betfair, Zynga, Blizzard Entertainment, Pokémon GO, Playtika, SGN

Telco

Comcast, verizon, KDDI, at&t, Viber, Telefónica

Financial Services

ADP, FICO, Wells Fargo, American Express, Concur, Citi, Generali, AXA, Visa

Digital Health

WebMD, BD, everyday HEALTH, Cochlear, Monsenso

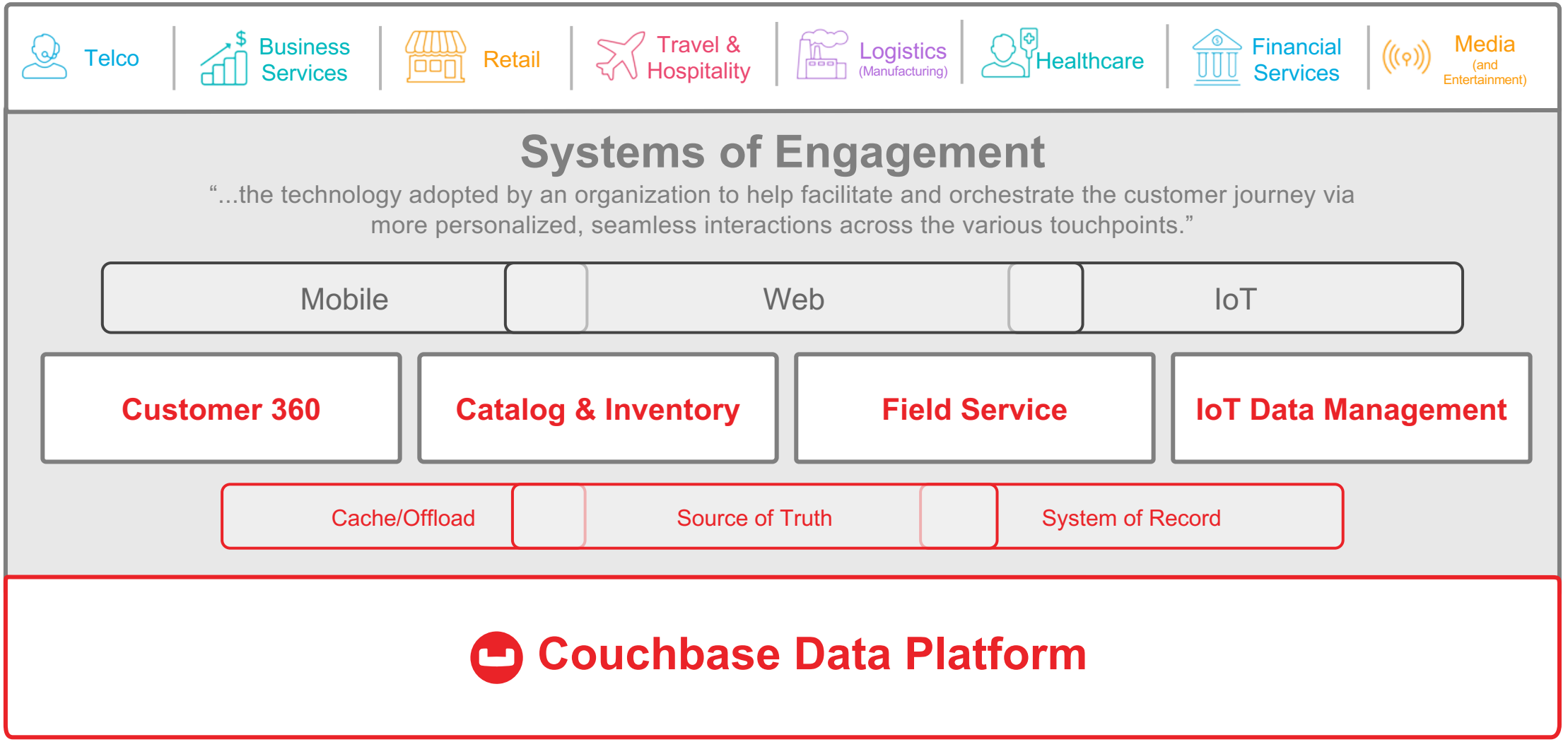
Digital Media

Gannett, Disney, McGraw Hill, Thomson Reuters, nielsen, Fandango, sky

Industrial IoT

Chamberlain, SUEZ environnement, GE, Numerex

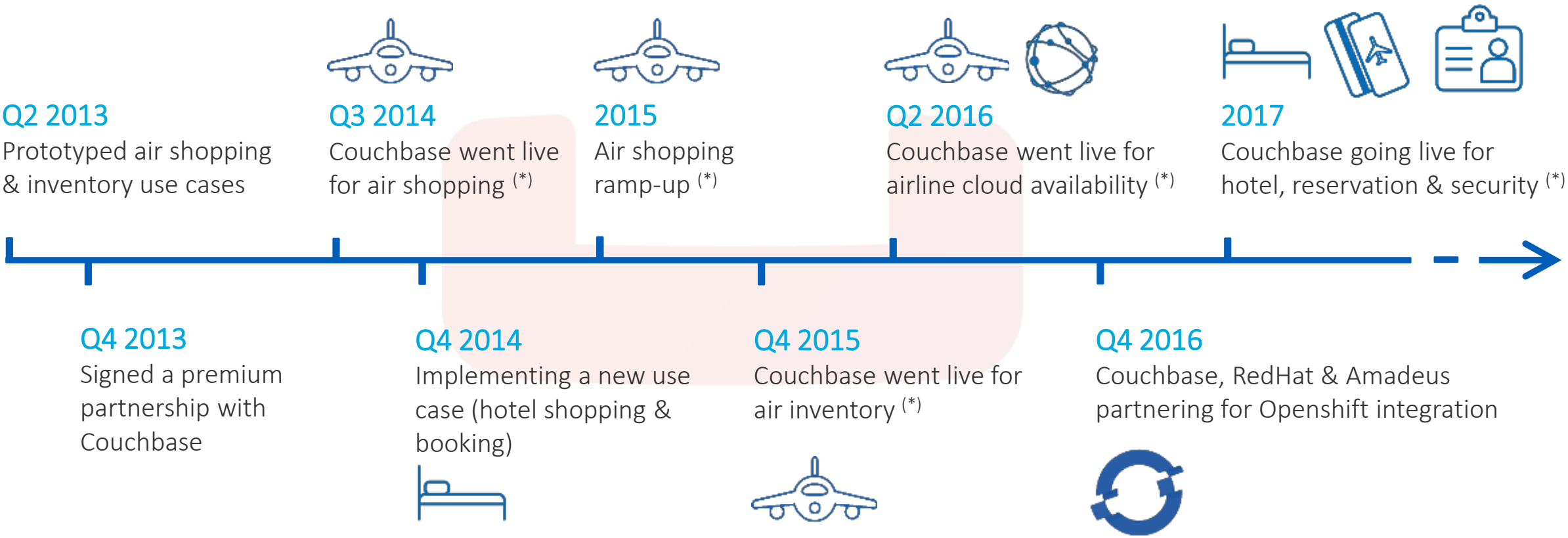
Where does Couchbase fit?



Source: Dynamic Yield <https://www.dynamicyield.com/glossary/systems-of-engagement/>

Couchbase & Amadeus

Partnering since 2013



Q2 2013
Prototyped air shopping & inventory use cases

Q3 2014
Couchbase went live for air shopping (*)

2015
Air shopping ramp-up (*)

Q2 2016
Couchbase went live for airline cloud availability (*)

2017
Couchbase going live for hotel, reservation & security (*)

Q4 2013
Signed a premium partnership with Couchbase

Q4 2014
Implementing a new use case (hotel shopping & booking)

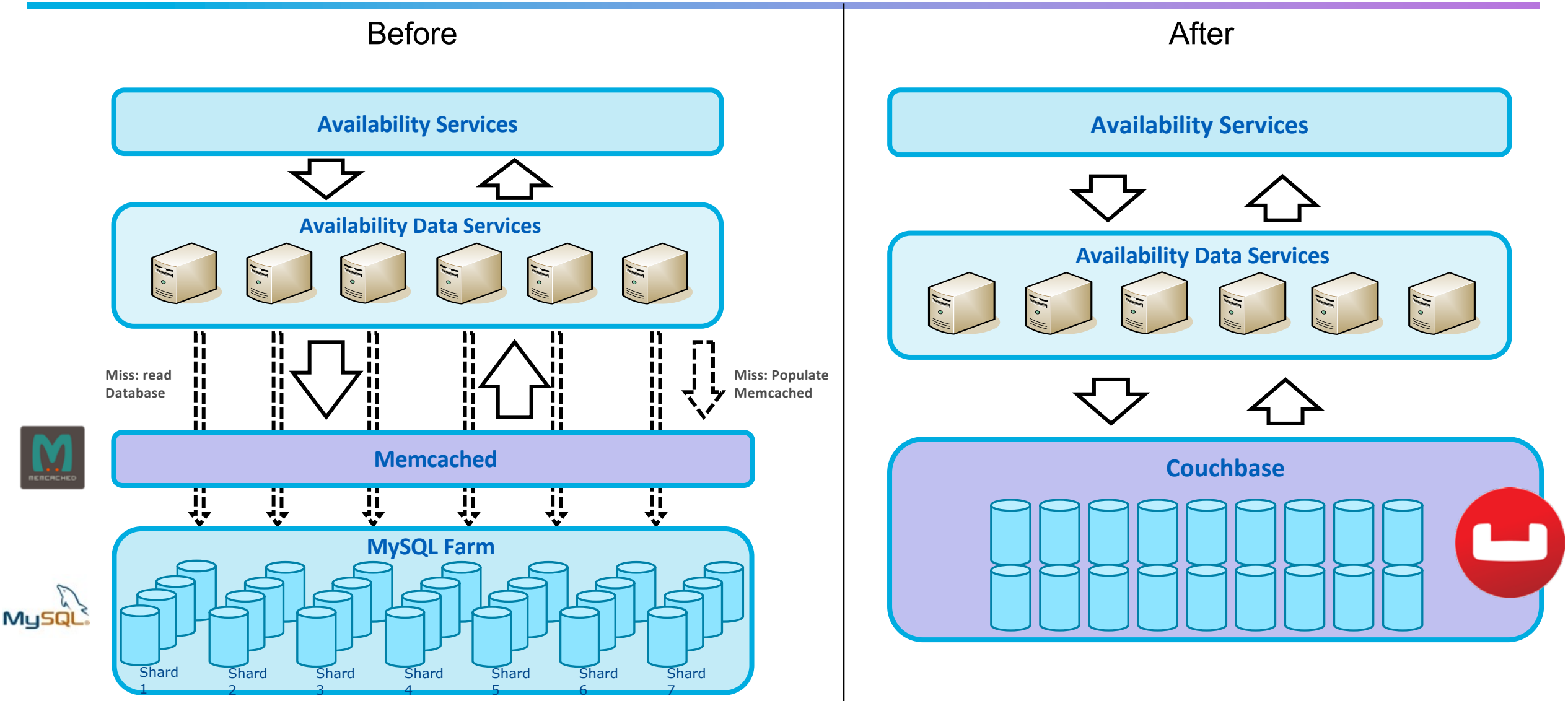
Q4 2015
Couchbase went live for air inventory (*)

Q4 2016
Couchbase, RedHat & Amadeus partnering for Openshift integration

Source: Amadeus presentation at Couchbase Europe 2017

(*) : for some functionalities.

Availability Cache Architecture: Before & After



Source: Amadeus presentation at Couchbase Connect 2015, Santa Clara CA

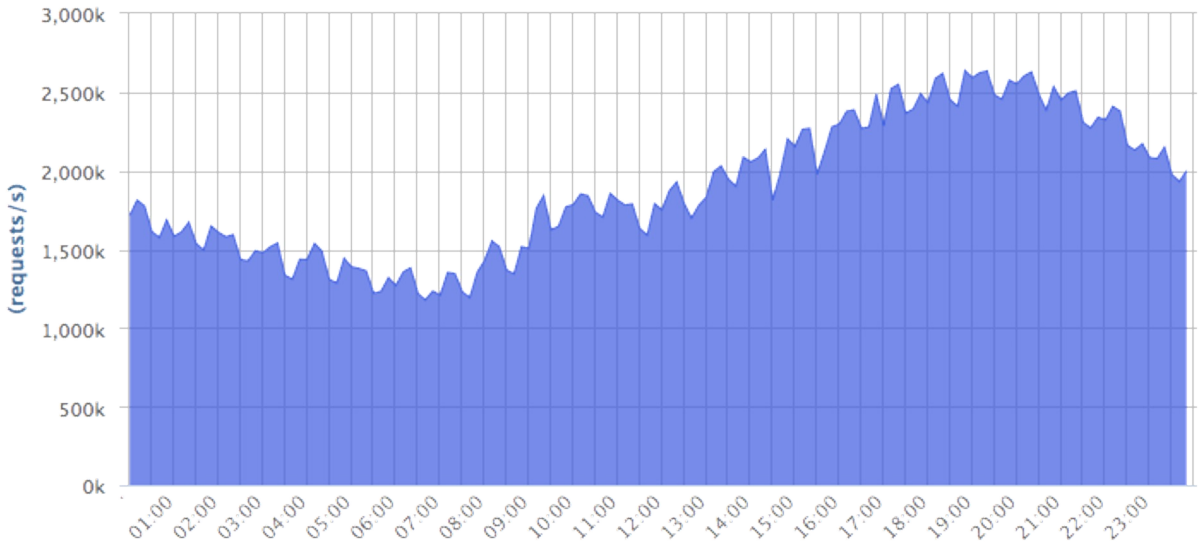
Problems Amadeus was trying to solve

- Simplifying the architecture to
  - Remove complex storage logic in the application layer
  - Decrease operational costs due to:
    - Inefficient hardware utilization
    - Administration overhead
- On-demand scaling because
  - Extending capacity is complex, requires preparation, and takes many days
- Eliminating downtime
  - Memcached outages rare, but very disruptive

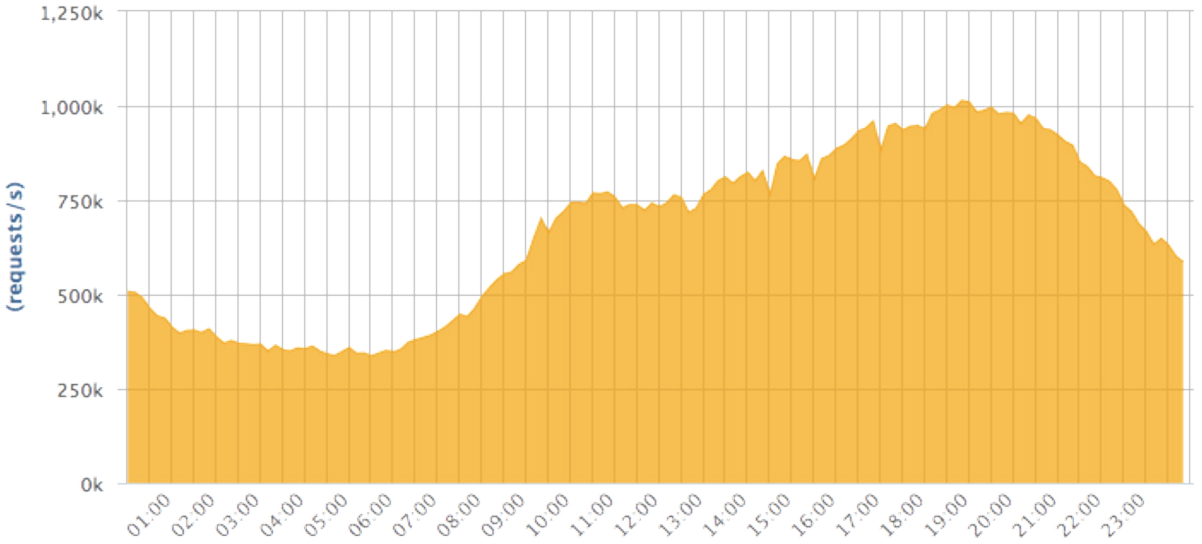
Source: Amadeus presentation at Couchbase Connect 2015, Santa Clara CA

Availability Data Workloads

GET 2.6 M/s



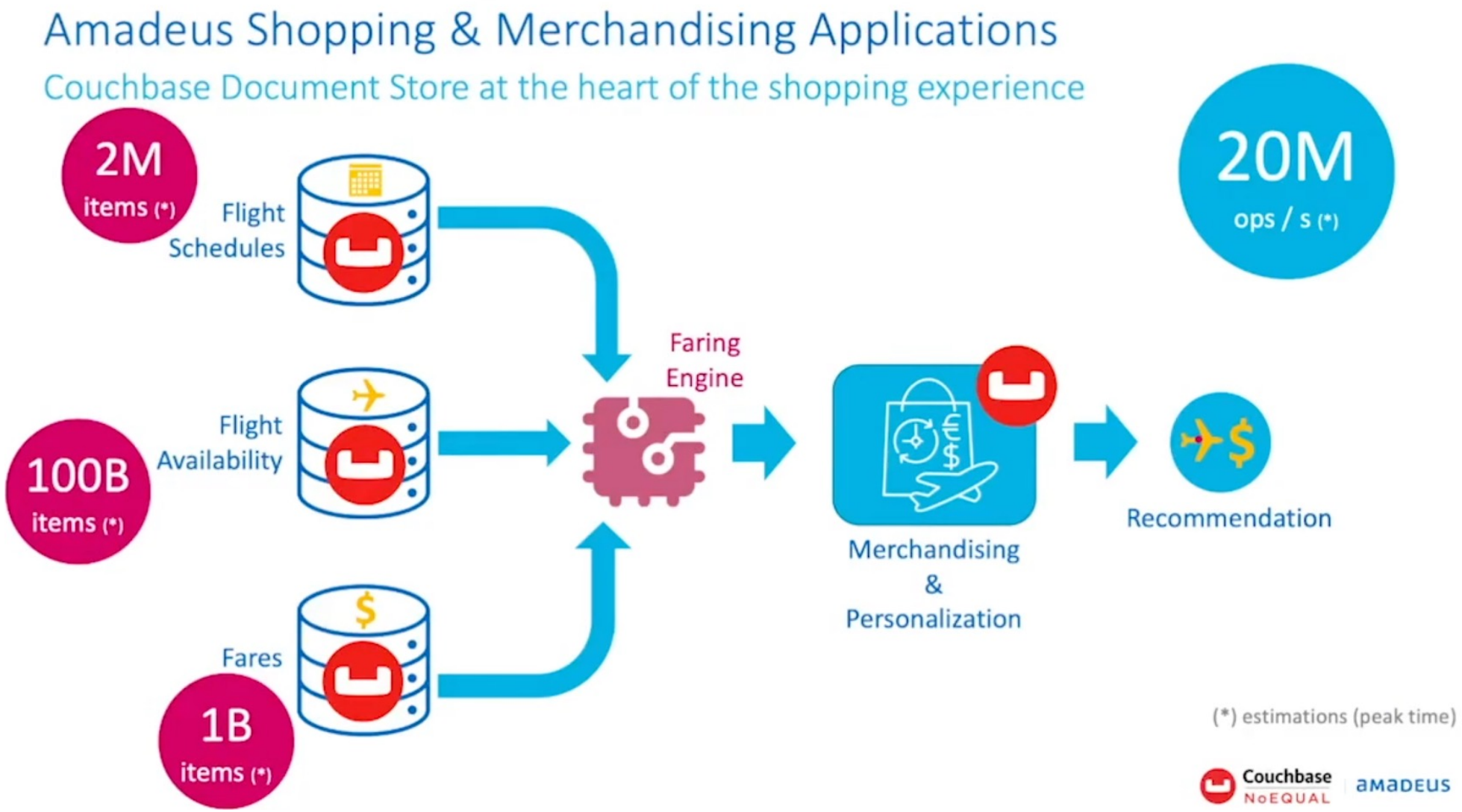
SET 1 M/s



- **Couchbase is fast**
  - Microseconds add up when performing millions of operations per second
  - Read operation latency < 0.5ms
- **Couchbase is *predictably* fast**
  - Long tail latencies would hurt application response time
  - Couchbase leverages large memory capacity
- **Couchbase online rebalancing works without impacting latency**

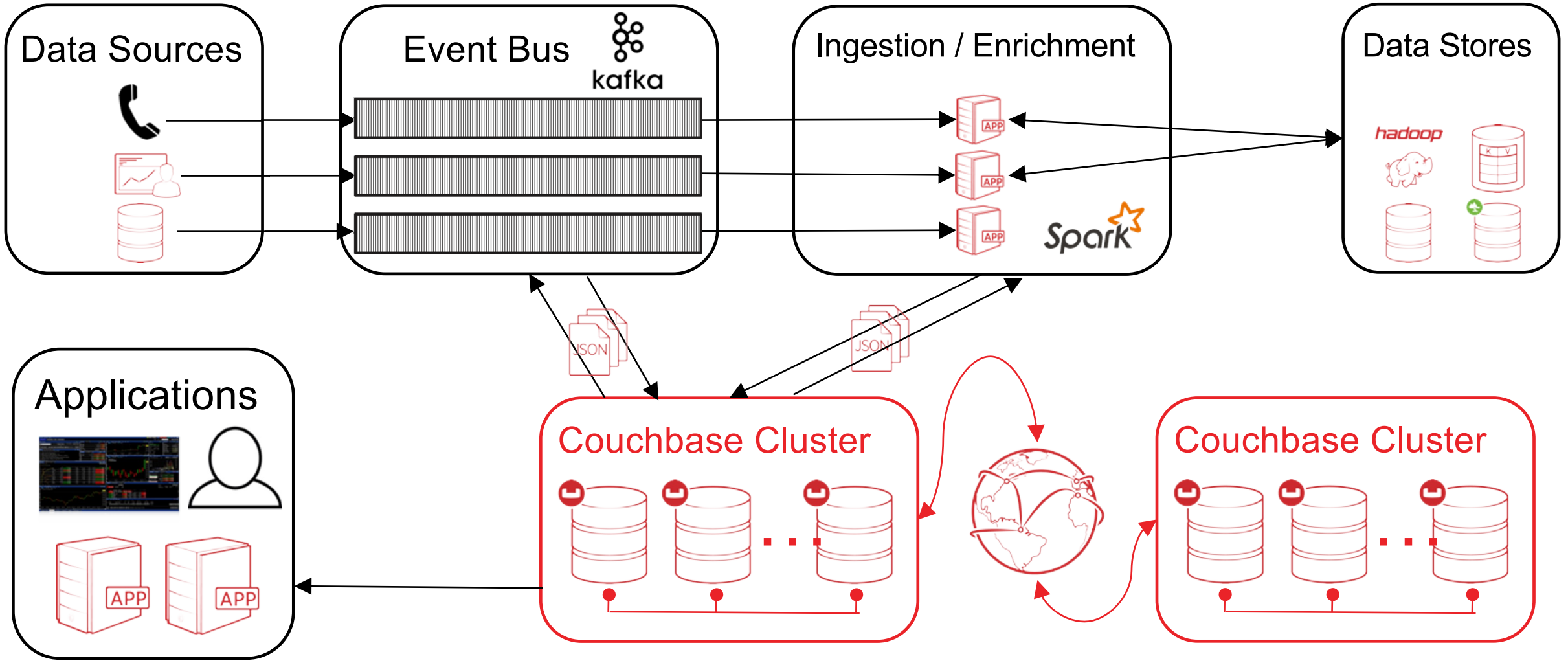
Source: Amadeus presentation at Couchbase Connect 2015, Santa Clara CA

Amadeus Shopping & Merchandising

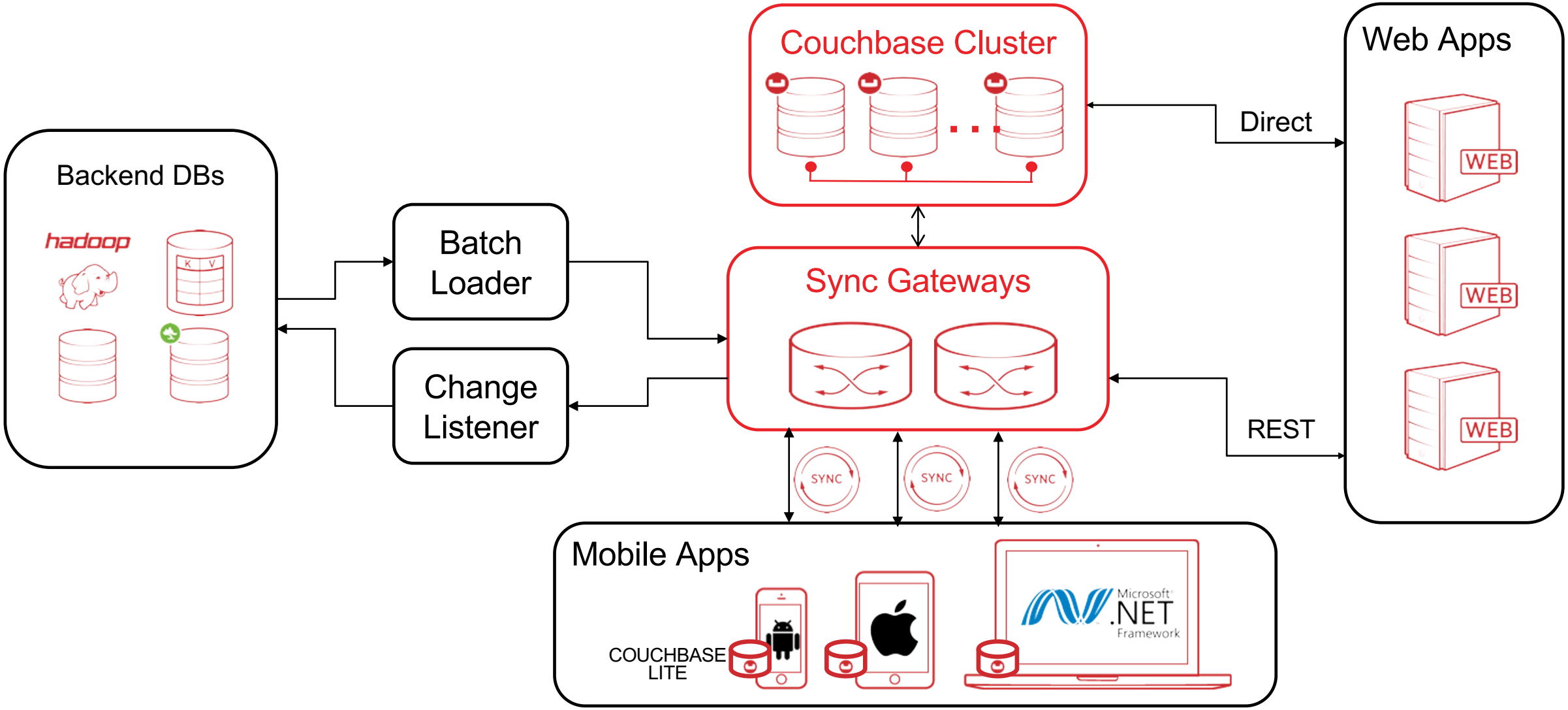


Source: Amadeus presentation at Couchbase Connect 2020

Reference Architectures: Streaming/Consolidated Data Store



Reference Architectures: Data Integration and Mobile Routing



THANK YOU

Couchbase