SPARK AND COUCHBASE: AUGMENTING THE OPERATIONAL DATABASE WITH SPARK

Will Gardella &

Matt Ingenthron

Couchbase

SPARK SUMMIT 2016

DATA SCIENCE AND ENGINEERING AT SCALE

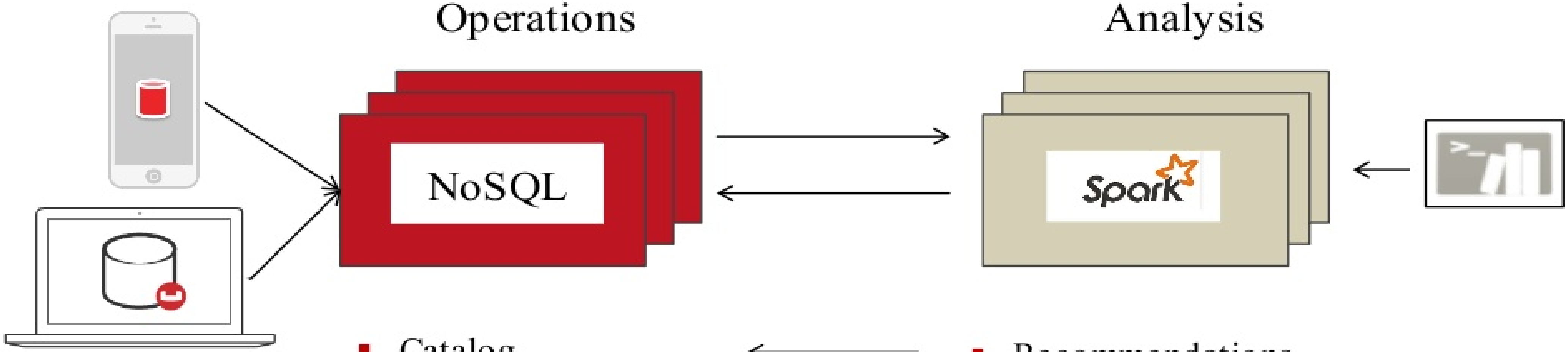
JUNE 6-8, 2016 SAN FRANCISCO

Agenda

- Why integrate Spark and NoSQL?
- Architectural alignment
- Integration "Points of Interest"
  - Automatic sharding and data locality
  - Streams: Data Replication and Spark Streaming
  - Predicate pushdown and global indexing
  - Flexible schemas and schema inference
- See it in action

WHY SPARK AND NOSQL?

NoSQL + Spark use cases



- Catalog
- Customer 360 + IOT
- Personalization
- Mobile applications

- Recommendations
- Next gen data warehousing
- Predictive analytics
- Fraud detection

Big Data at a Glance

OPERATIONAL ANALYTICAL

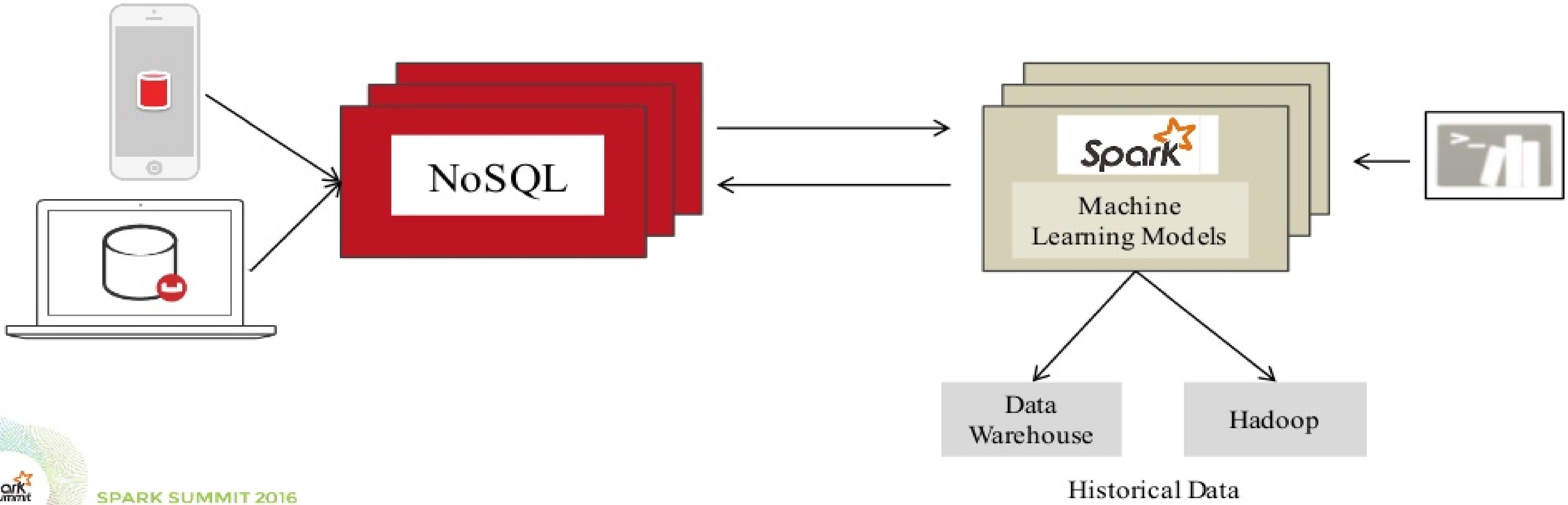
	Couchbase	Spark	Hadoop
Use cases	• Operational • Web / Mobile	• Analytics • Machine Learning	• Analytics • Machine Learning
Processing mode	• Online • Ad Hoc	• Ad Hoc • Batch • Streaming (+/-)	• Batch • Ad Hoc (+/-)
Low latency =	< 1 ms ops	Seconds	Minutes
Performance	Highly predictable	Variable	Variable
Users are typically…	Millions of customers	100's of analysts or data scientists	100's of analysts or data scientists
	Memory-centric	Memory-centric	Disk-centric
Big data =	10s of Terabytes	Petabytes	Petabytes

Use Case: Operationalize Analytics / ML

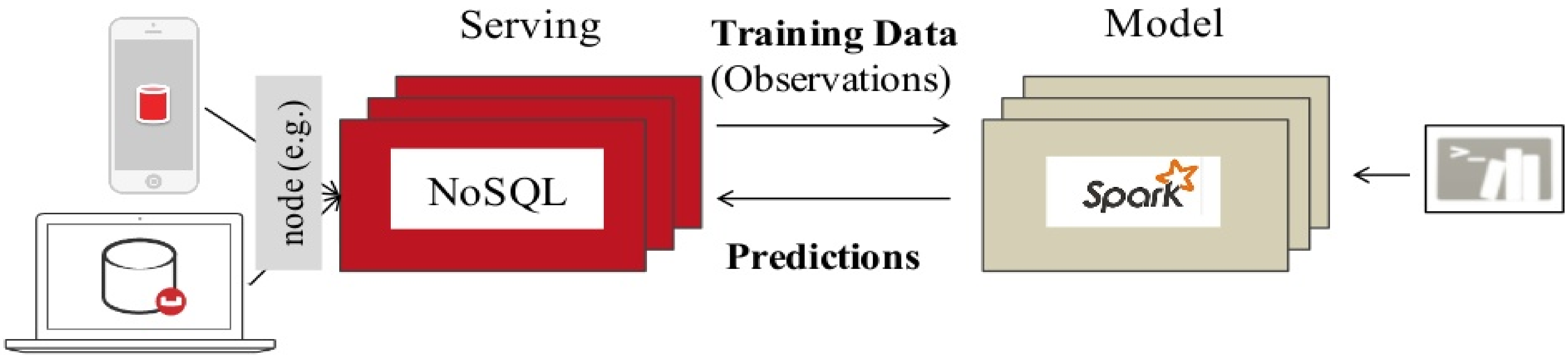
Examples: recommend content and products, spot fraud or spam

Data scientists train machine learning models

Load results into Couchbase so end users can interact with them online



Use Case: Operationalize ML



Why NoSQL with Spark?

	RDBMS Challenges	NoSQL Strengths
Scaling	Hard	Easy
Sharding & replication	Manual	Automatic
XDCR, geo distro, disaster recovery	Difficult, expensive	Easy, performant
Performance	Add cache	Integrated cache
Agility	Schema migrations	Flexible data model
Upgrades & maintenance	Downtime	Online
Cost	$$$	$

Adapted from: Databricks – Not Your Father's Database https://www.brighttalk.com/webcast/12891/196891

Spark connects to everything…

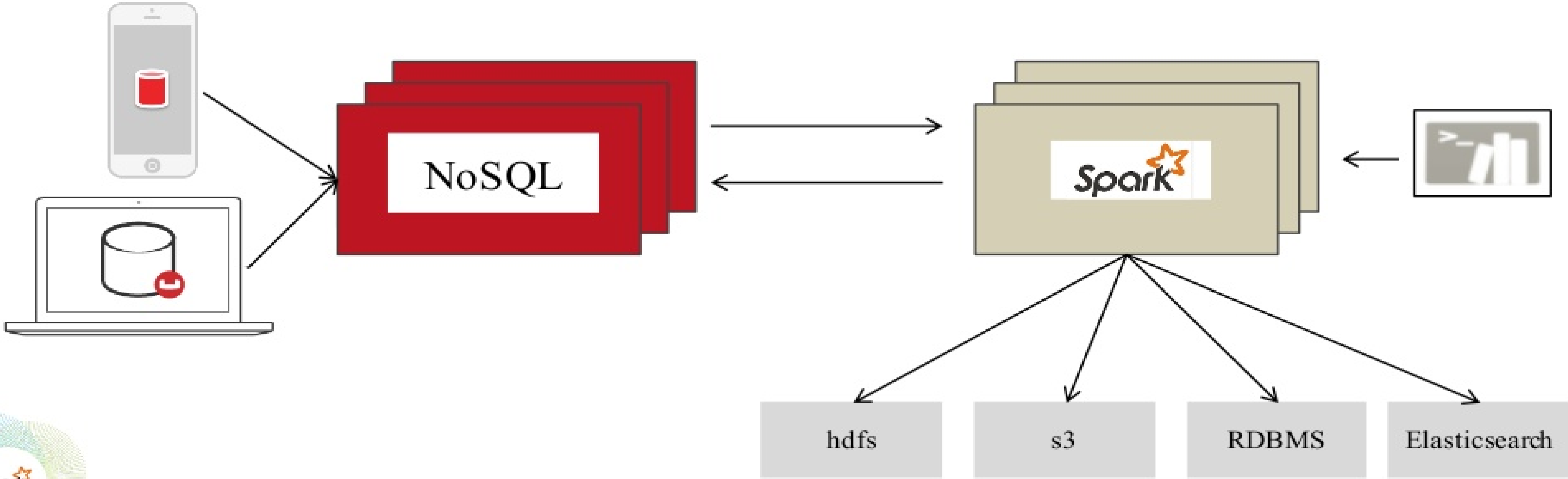
databricks

Use Case #2: Data Integration

Data engineers query data in many systems w/ one language & runtime

Store results where needed for further use

Late binding of schemas



ARCHITECTURAL ALIGNMENT

Key-Value

Directly fetch / store a particular record

Query

Specify a set of criteria to retrieve relevant data records.
Essential in reporting.

Map-Reduce Views

Maintain materialized indexes of data records, with reduce functions for aggregation

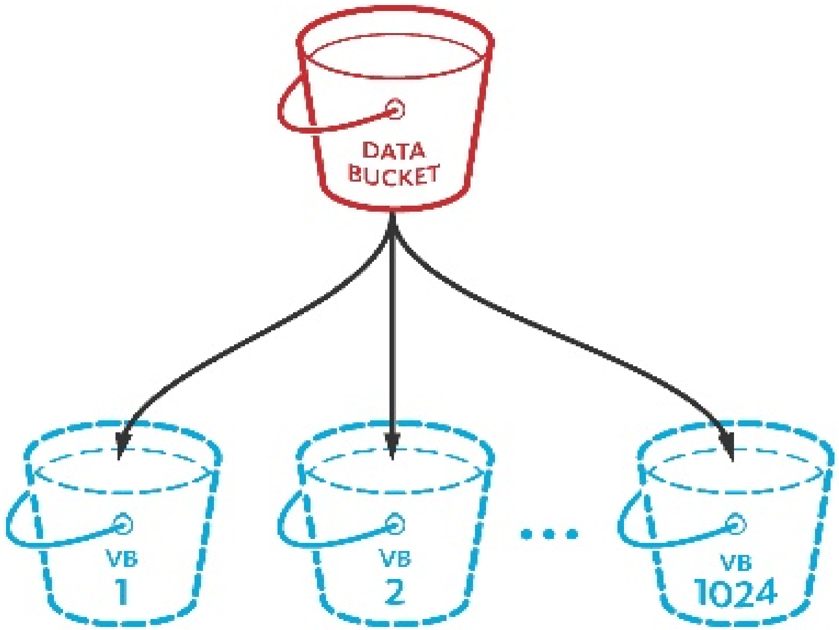
Data Streaming

Efficiently, quickly stream data records to external systems for further processing or integration

Full Text Search

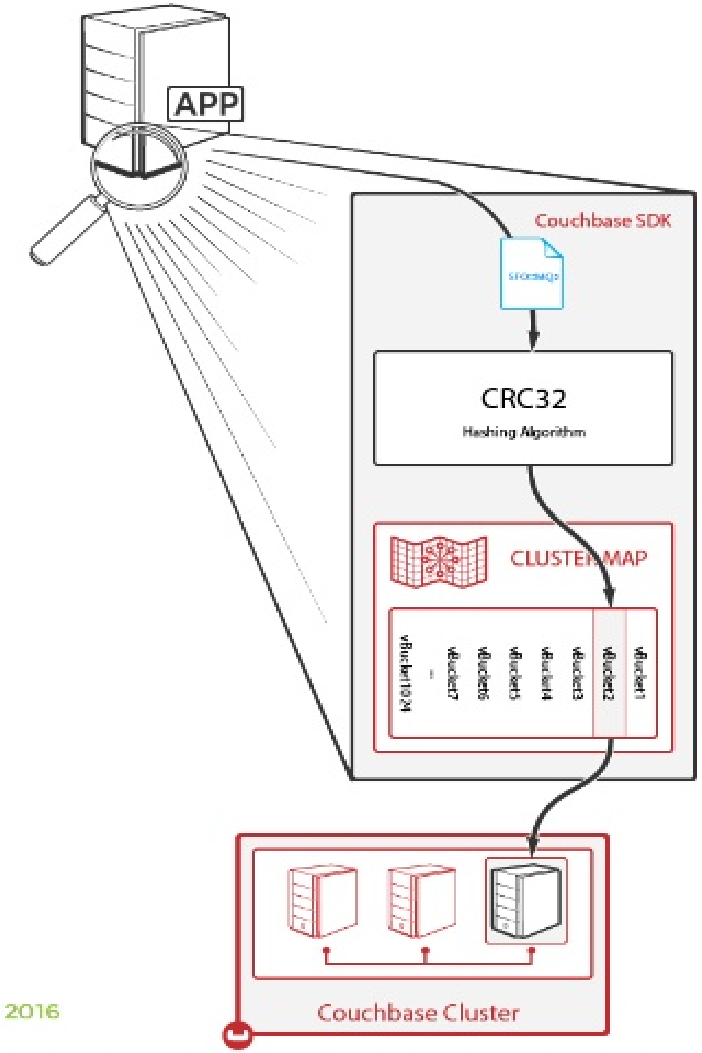
Search for and fetch the most relevant records given a freeform text string

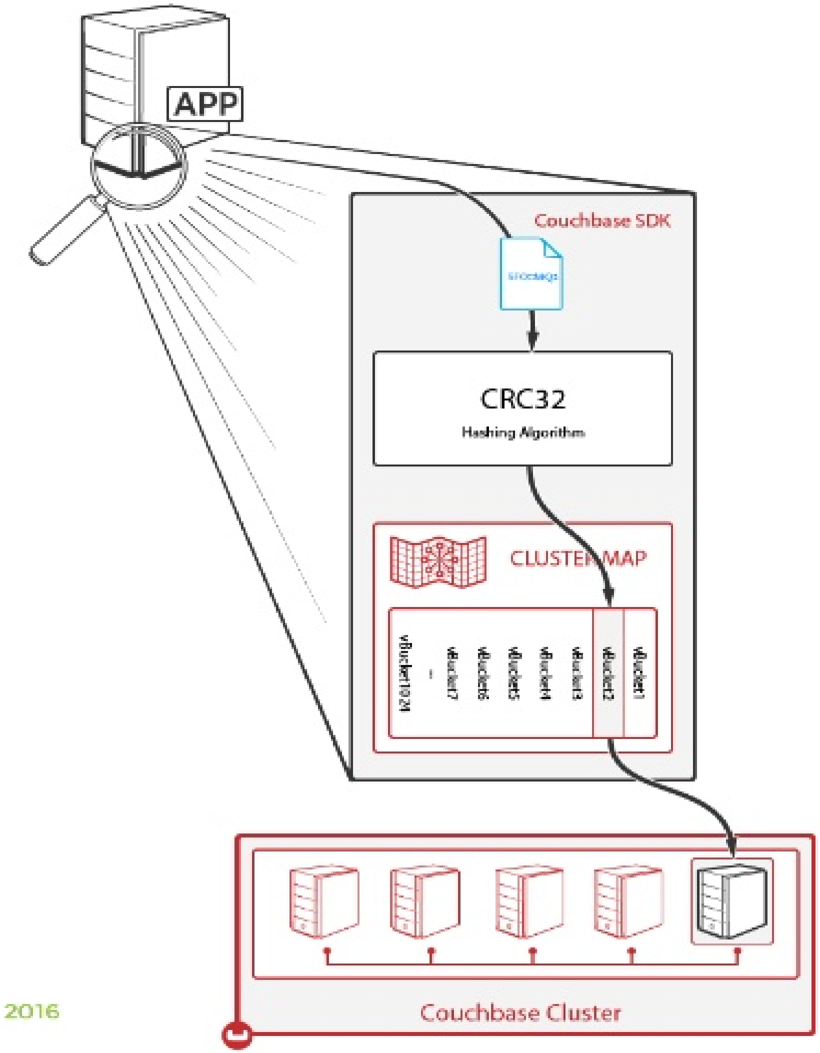
Hash Partitioned Data



Auto Sharding – Bucket And vBuckets

- A bucket is a logical, unique key space
- Each bucket has active & replica data sets
  - Each data set has 1024 virtual buckets (vBuckets)
  - Each vBucket contains 1/1024th of the data set
  - vBuckets have no fixed physical server location
- Mapping of vBuckets to physical servers is called the cluster map
- Document IDs (keys) always get hashed to the same vBucket
- Couchbase SDK's lookup the vBucket → server mapping


APP
Couchbase SDK
CRC32
Hashing Algorithm
CLUSTER MAP
vBucket1
vBucket2
vBucket3
vBucket4
vBucket5
vBucket6
vBucket7
...
vBucket1024
Couchbase Cluster


APP
Couchbase SDK
CRC32
Hashing Algorithm
CLUSTER MAP
vBucket1
vBucket2
vBucket3
vBucket4
vBucket5
vBucket6
vBucket7
...
vBucket1024
Couchbase Cluster

N1QL Query

- N1QL, pronounced "nickel", is a SQL service with extensions specifically for JSON
  - Is stateless execution, however…
  - Uses Couchbase's Global Secondary Indexes.
    - These are sorted structures, range partitioned.
  - Both can run on any nodes within the cluster. Nodes with differing services can be added and removed as needed.

MapReduce Couchbase Views

- A JavaScript based, incremental Map-Reduce service for incrementally building sorted B+Trees.
  - Runs on every node, local to the data on that node, stored locally.
  - Automatically merge-sorted at query time.

Data Streaming with DCP

- A general data streaming service, Database Change Protocol.
  - Allows for streaming all data out and continuing, or…
  - Stream just what is coming in at the time of connection, or…
  - Stream everything out for transfer/takeover…

COUCHBASE FROM SPARK

Key-Value

Direct f...

Produce and store RDDs in Spark programs

Query

Specifying ... crit...

Use Spark SQL for accessing Couchbase

Map-Reduce Views

Maintain ... i... f... ag...

Query Couchbase for view results as RDDs

Data Streaming

Effici...

Expose data streams through the Spark DStream interface

Full Text Search

Search for, and allow tuning of the system to fetch the most relevant records given a freeform search string.

Integration Points of Interest

AUTOMATIC SHARDING AND DATA LOCALITY

What happens in Spark Couchbase KV

- When 1 Spark node per CB node, the connector will use the cluster map and push down location hints
  - Helpful for situations where processing is intense, like transformation
  - Uses pipeline IO optimization
- However, not available for N1QL or Views
  - Round robin - can't give location hints
  - Back end is scatter gather with 1 node responding

Integration Points of Interest

PREDICATE PUSHDOWN AND GLOBAL INDEXING

SparkSQL on
N1QL with Global Secondary Indexes

TableScan

Scan all of the data and return it

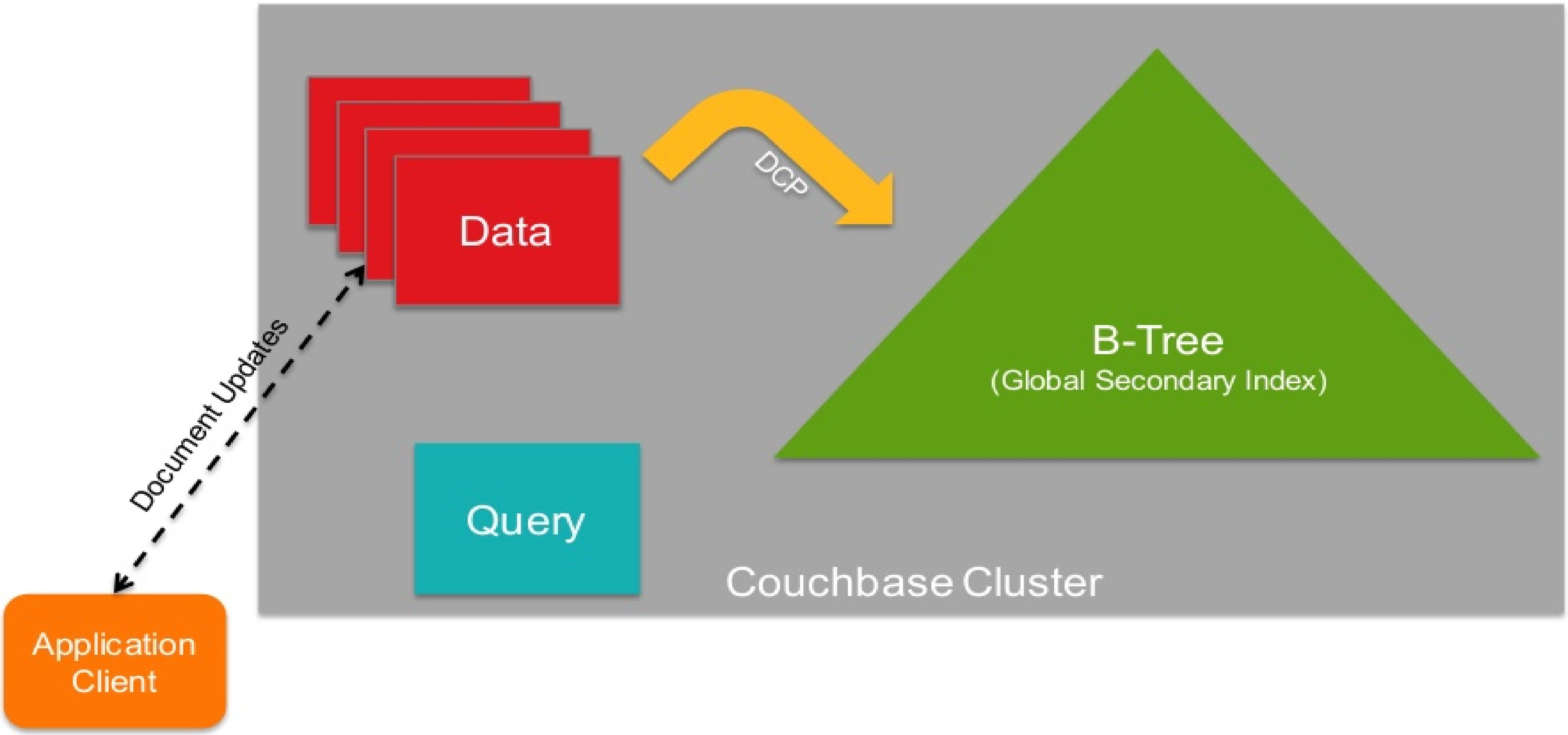
PrunedScan

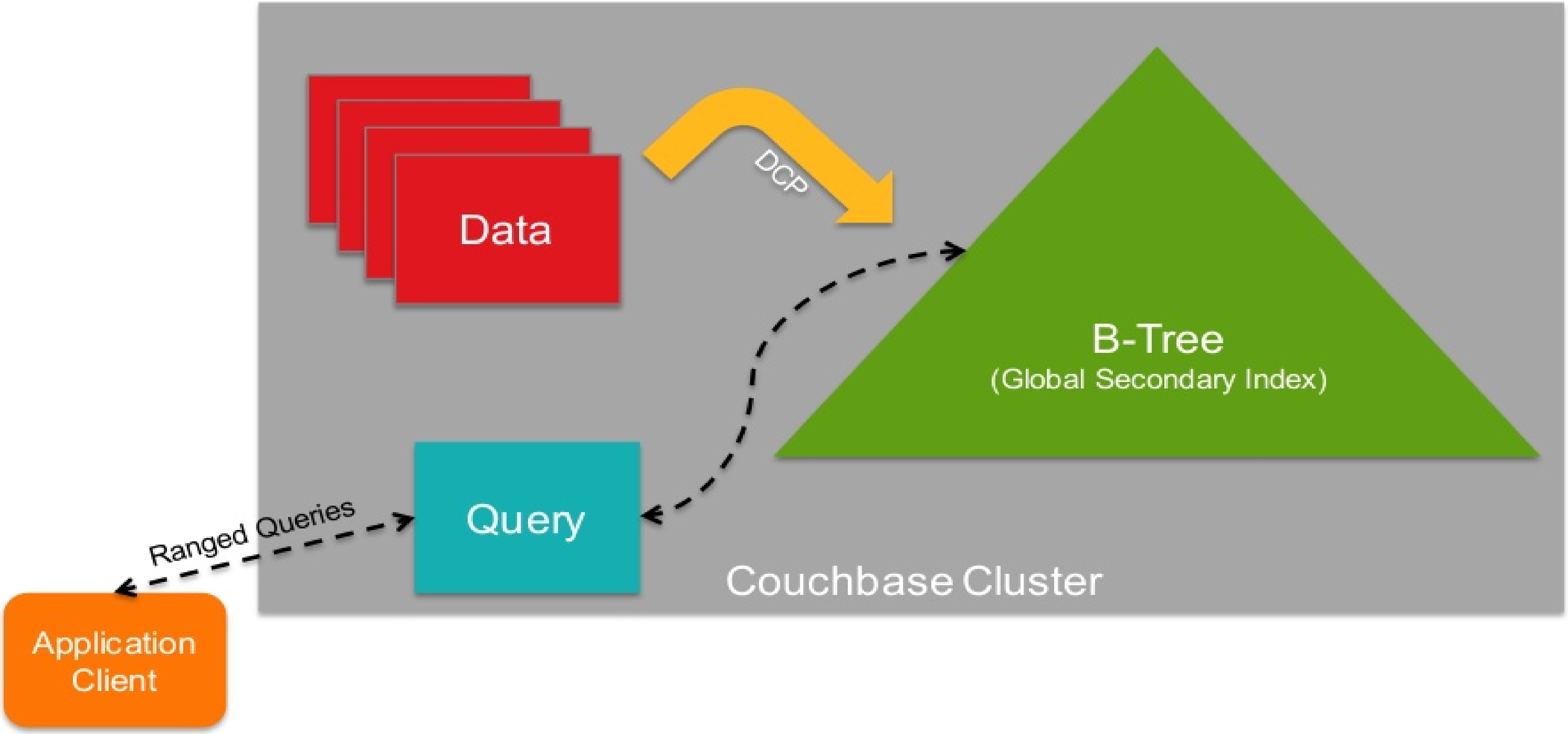
Scan an index that matches only r
to the query at hand.

PrunedFilteredScan

Scan an index that matches only relevant da
to the query at hand.

Couchbase's connector implements a PrunedFilteredScan which passes through the Couchbase Query optimizer ensuring highest efficiency and minimal data transfer.

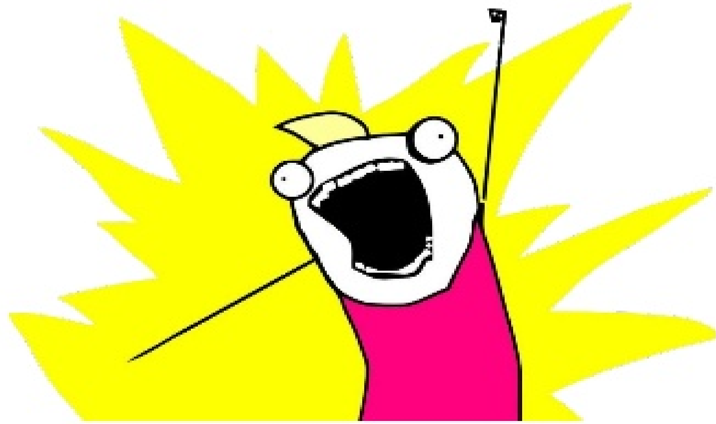

Data
DCP
B-Tree
(Global Secondary Index)
Document Updates
Query
Couchbase Cluster
Application Client


Data
DCP
B-Tree
(Global Secondary Index)
Query
Ranged Queries
Couchbase Cluster
Application
Client

Predicate pushdown

```
def filterToExpression(filter: Filter): String = {
  filter match {
    case EqualTo(attr, value) => s" `$attr` = " + valueToFilter(value)
    case GreaterThan(attr, value) => s" `$attr` > " + valueToFilter(value)
    case GreaterThanOrEqual(attr, value) => s" `$attr` >= " + valueToFilter(value)
    case LessThan(attr, value) => s" `$attr` < " + valueToFilter(value)
    case LessThanOrEqual(attr, value) => s" `$attr` <= " + valueToFilter(value)
    case IsNull(attr) => s" `$attr` IS NULL"
    case IsNotNull(attr) => s" `$attr` IS NOT NULL"
    case StringContains(attr, value) => s" CONTAINS(`$attr`, '$value')"
    case StringStartsWith(attr, value) => s" `$attr` LIKE '$value%'"
    case StringEndsWith(attr, value) => s" `$attr` LIKE '%$value'"
    case In(attr, values) => {
      val encoded = values.map(valueToFilter).mkString(",")
      s" `$attr` IN [$encoded]"
    }
```

Predicate pushdown



Notes from implementing:

- Spark assumes it's getting all the data, applies the predicates

Future potential optimizations

- Push down all the things!
  - Aggregations
  - JOINs
- Looking at Catalyst engine extensions from SAP
  - But, it's not backward compatible and…
  - …many data sources can only push down filters

image courtesy http://allthefreethings.com/about/

Integration Points of Interest

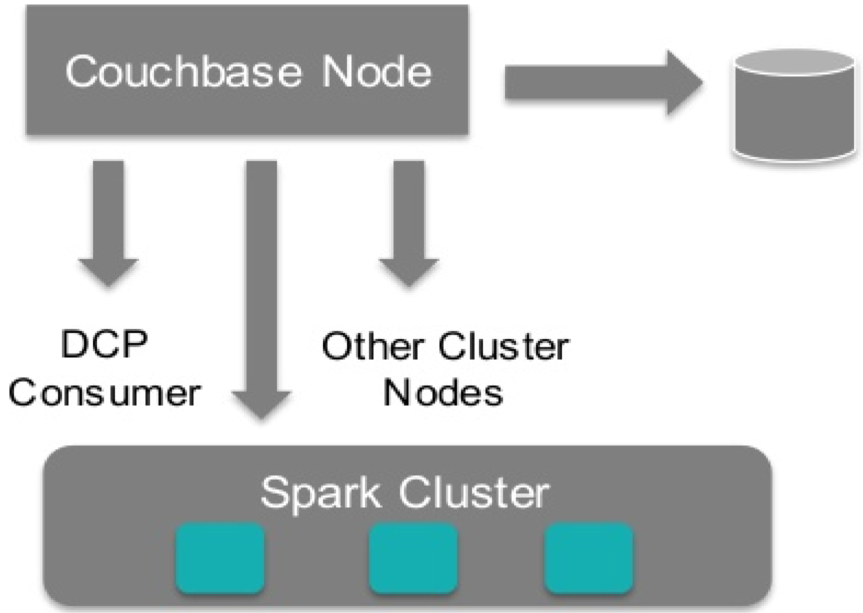
STREAMS: DATA REPLICATION AND SPARK STREAMING

DCP and Spark Streaming

- Many system architectures rely upon streaming from the ‘operational’ data store to other systems
  - Lambda architecture => store everything and process/reprocess everything based on access
  - Command Query Responsibility Segregation - (CQRS)
  - Other reactive pattern derived systems and frameworks

DCP and Spark Streaming

- Documents flow into the system from outside
- Documents are then streamed down to consumers
- In most common cases, flows memory to memory



SEE IT IN ACTION

Couchbase Spark Connector 1.2

- Spark 1.6 support, including Datasets
- Full DCP flow control support
- Enhanced Java APIs
- Bug fixes

QUESTIONS?

THANK YOU.

@ingenthr & @willgardella

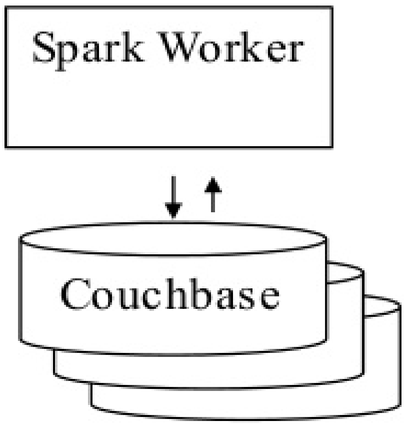
Try Couchbase Spark Connector 1.2

http://www.couchbase.com/bigdata

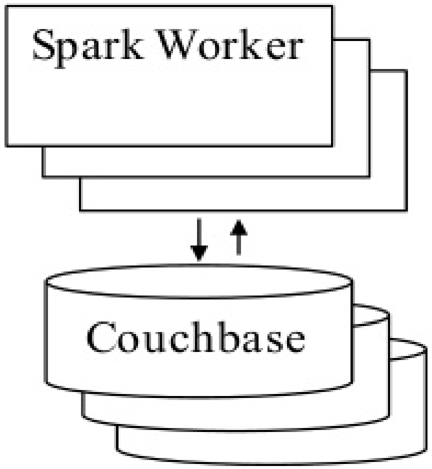
SPARK SUMMIT 2016
DATA SCIENCE AND ENGINEERING AT SCALE
JUNE 6-8, 2016 SAN FRANCISCO

ADDITIONAL INFORMATION

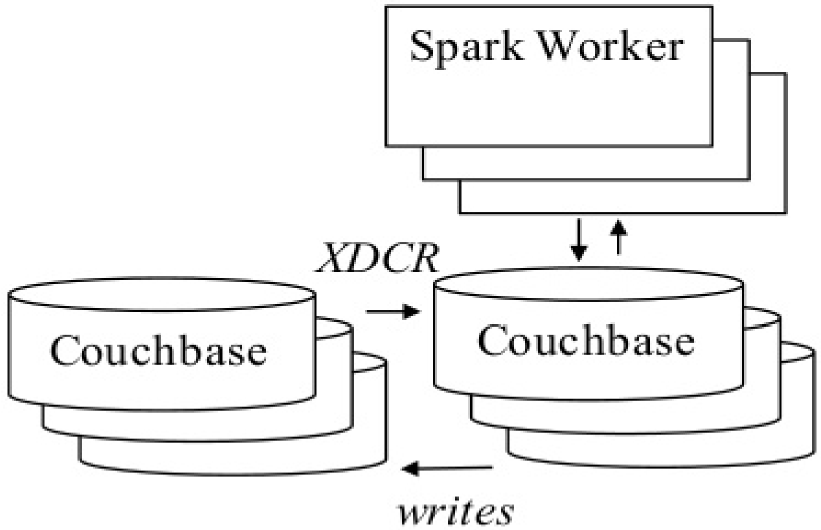
Deployment Topology



- Many small gets
- Streaming with low mutation rate
- Ad hoc



- Medium processing
- Predictable workloads
- Plenty of overhead on machines



- Heaviest processing
- Workload isolation