

ORACLE®

# Big Data SQL and Query Franchising

An Architecture for Query Beyond Hadoop

Dan McClary, Ph.D.  
Big Data Product Management  
Oracle



# Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

# Agenda

- 1 Fixing the fashion problem
- 2 Why Unified Query Matters
- 3 Query Franchising: an architecture for unified query
- 4 Customer 360 Live!

# “Oracle has a fashion problem”

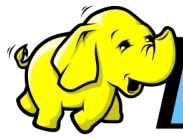
Every industry analyst I’ve ever talked with.

We thought everything went with black.



# Risk Removal, Not Empire Building

- Make the Big Data ecosystem easy to consume
  - Simplified operations
  - Databricks' certified Spark environment
  - Cloudera's Enterprise Data Hub
- Encourage and support the use of new technologies
  - We build on Hadoop!
  - You should build on Hadoop!
- De-risk new innovations with bridges to the business



# *hadoop* @ Oracle Global Support Services



- Cloudera's Distribution captures HW failures
- Integrate with customer telemetry, configurations, service history, diagnostics
- **Predict failures, save money, and serve customer better**

Anticipate

Detect

Predict

Automate

Delight



# Oracle Big Data Discovery: Built on *Spark*

The interface is divided into several sections:

- Top Left:** Oracle Big Data Discovery header, search bar (chicag), and navigation for 9 Projects and 89 data sets.
- Top Center:** Data Sets Refinements and Data Sets list, including ChicagoCensus2011 and ChicagoCrime2011.
- Top Right:** Overview of 1M records sampled, 1M records viewed, and 74 attributes. Includes charts for Date reviewed, Score, Region, Price, and Zipcode.
- Bottom Left:** Sales Analysis: Customer Churn. A bubble chart showing Record Count vs Record Count by p\_body.
- Bottom Right:** Data table with columns: #\_dateviewed, #\_p\_name, #\_p\_year, #\_p\_description, #\_p\_drinkability, #\_p\_winery, #\_p\_region, #\_p\_price.

**Central Diagram:** A circular process flow with four quadrants: Find, Explore, Transform, and Discover, surrounding a central icon of a person at a computer.



# Oracle Data Enrichment Cloud Service: Uses **Spark**

**ORACLE Data Enrichment Cloud Service**

Catalog | Dashboard | Policies | Documentation

< Catalog | Kevin65K

Refresh | Reset | Undo | Redo | Recommendations

Search

**Transform Script**

- Rename Col\_0021 to date\_02
- Rename Col\_0022 to us\_san
- Rename Col\_0023 to occupation
- Rename Col\_0001 to identifier
- Rename Col\_0005 to middle\_initial
- Extract area\_code from us\_phone
- Enrich column city with city.country
- Enrich column city with city.population

**Recommendations for city**

- Enrich column city with city.state
- Enrich column city with city.lat
- Enrich column city with city.lon

Column	Type	Sample Values
identifier	number	1030489; 1020420; 1040779; 1021076; 1011915; 1053392
Col_0002	text	male; female
name_title	text	Mr.; Ms.; Mrs.; Dr.
first_name	text	James; John; Robert; Mary; Michael; William; David; Richard
middle_initial	text	J; M; R; D; C; A; S; L; B; E
last_name	text	Smith; Johnson; Williams; Jones; Brown; Davis; Miller; Moore
Col_0007	text	84 Henry Ford Avenue; 2966 Garfield Road; 1024 Burnside Beechwood Avenue; 3084 Windy Ridge Road; 2375 Zimmerman
city	text	New York; Los Angeles; Chicago; Houston; Philadelphia; Dallas
population	text	8413351; 3857883; 2868594; 2275108; 1446025; 1292773
county	text	Los Angeles; Cook; Harris; Philadelphia; Dallas; Fulton; San Diego
state	text	CA; TX; NY; FL; IL; PA; MI; OH; NJ; MA
zipcode	number	90017; 19108; 49503; 48075; 30303; 94612; 10011; 10013
Col_0011	text	US
Col_0012	text	United States
email	text	CharlesCDilworth@cuvov.de; JamieEWilson@jourrapide.com; LaurieMHavens@dayrep.com; ClarenceBRogers@teleworm.us; EileenREvans@qustr.com; MarthaTCoffett@teleworm.us; J

**Column Profile Results**

Metric	Count	Percent
Total Rows in File	65538	100%
Rows with Data	65538	100%
Rows with No Data (null)	2	0%
Distinct Values in Column	7319	11%
Duplicate Values in Col.	58217	89%
Distinct Patterns	217	0%

Detected Type: TEXT  
Subtype: city  
Matching Domain: city  
Match Accuracy: 95%  
Invalid Values: 5%

Scalable Machine Learning Pipeline  
Natural Language Processing  
Knowledge Graph-driven Repair and Blending

Interactive Recommendations and  
Visual Profiling to Transform Noisy Signal  
Into Refined Information Sets for Analytics

**ORACLE Data Enrichment Cloud Service**

Catalog | Dashboard | Policies | Documentation

COMPARE ALL  
Last updated Sep 17, 2014

Transformations: **49.8 M**

Jobs Running: **0**

Rows Processed: **4.2 M**

Warnings: **0**

Errors: **0**

24H 72H 7D 30D

**File Types**

**File Sizes**

**File #Rows**

Current Jobs

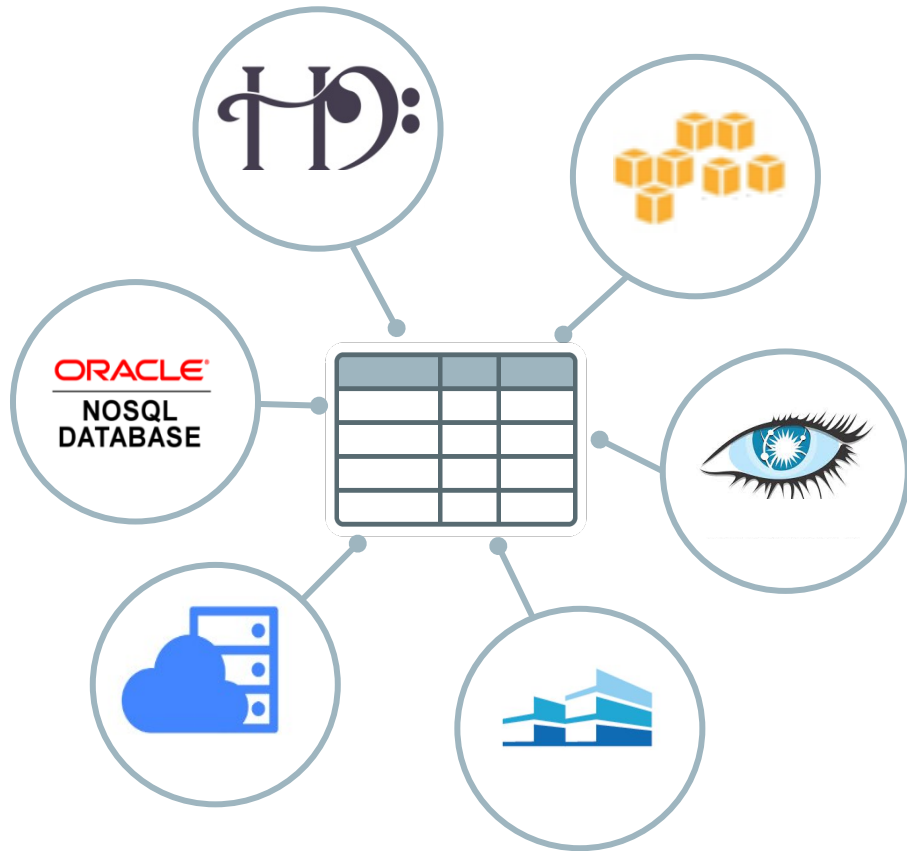
Job Status	Count
Completed	78
Pending	148
Running	0

# Data Analytics Challenge (2012)

Separate data access interfaces



# Tables on NoSQL



# SQL on Hadoop



# Data Analytics Challenge: 2013

No comprehensive SQL interface across Oracle, Hadoop and NoSQL

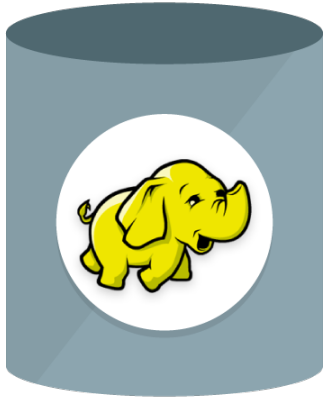
CQL   find()   UnQL



SQL

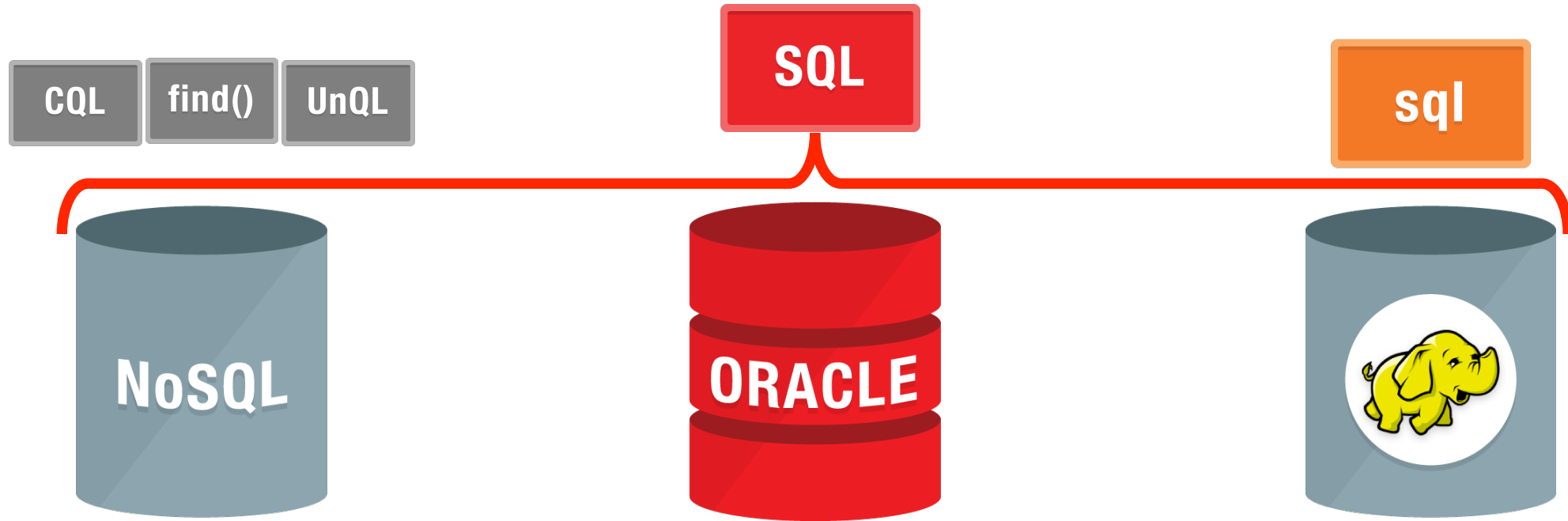


sql



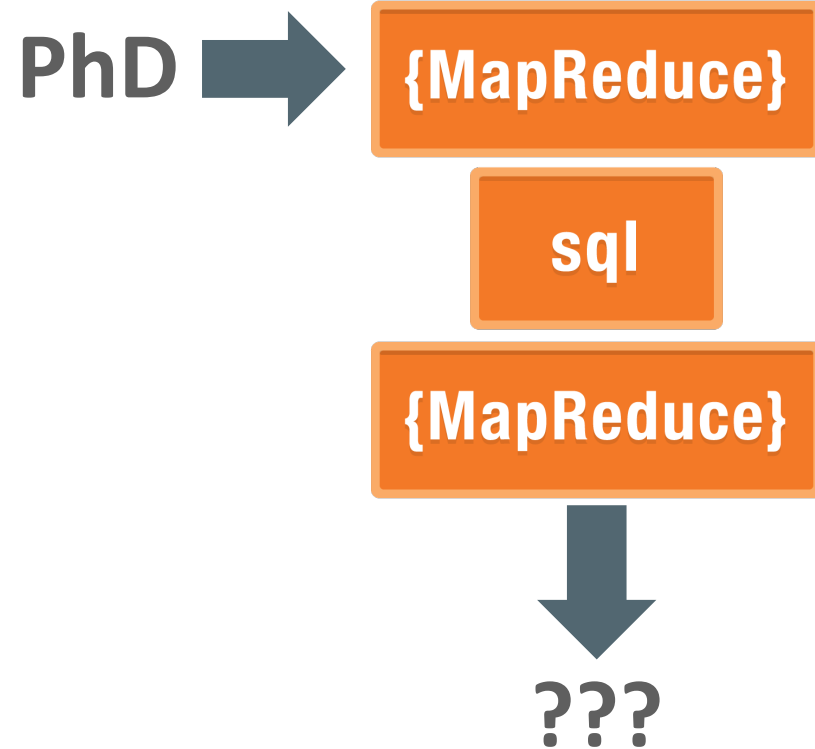
# Oracle Big Data Management System

Unified SQL access to all enterprise data



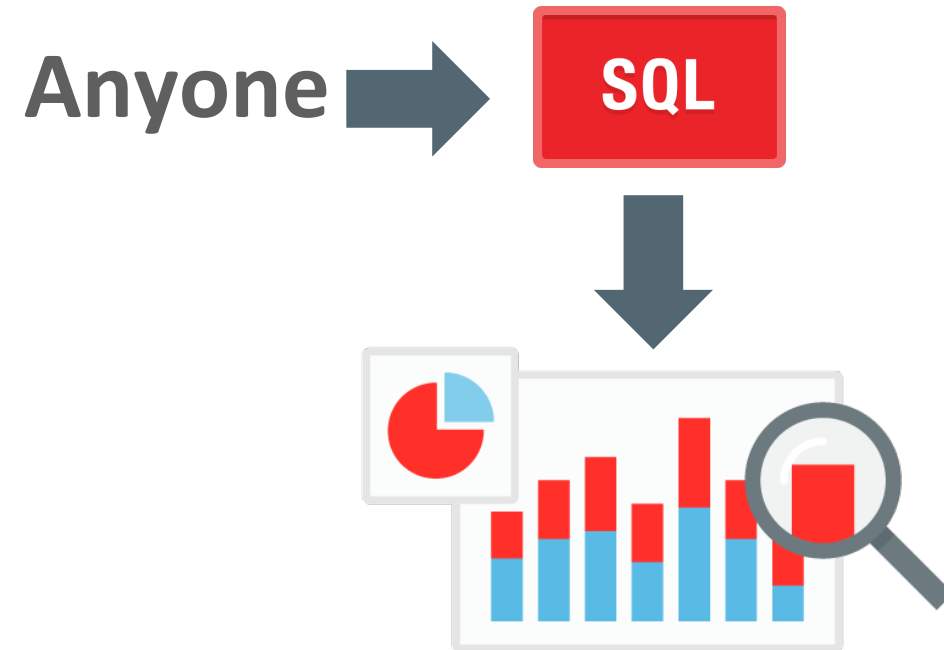
# What Does Unified Query Mean for You?

Before

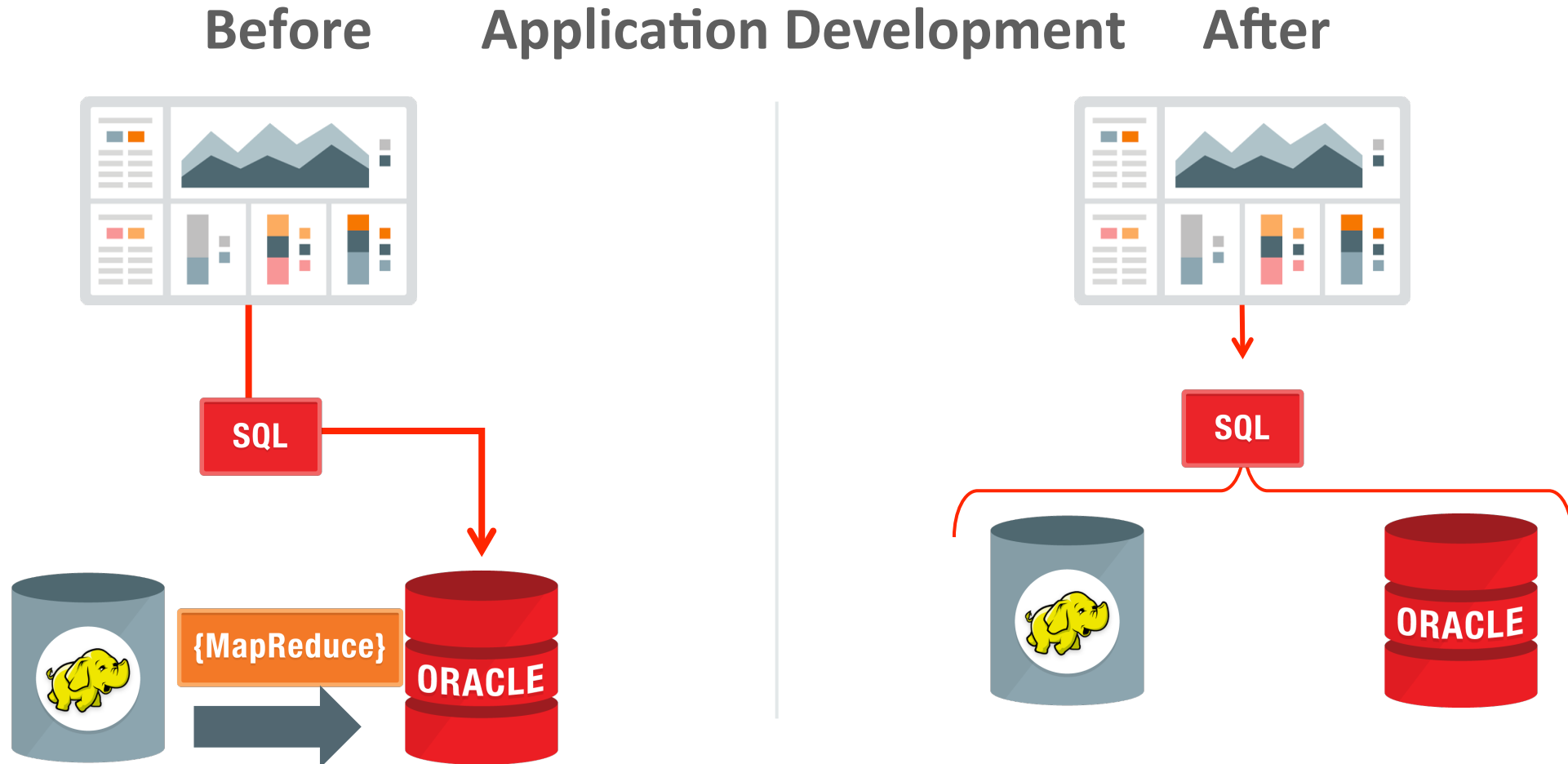


Data Science

After



# What Does Unified Query Mean for You?





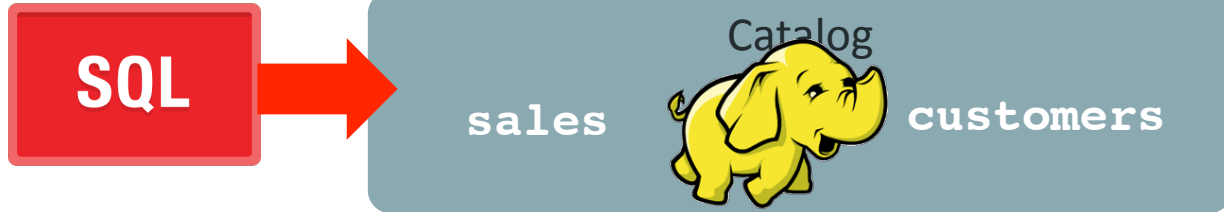
# How Does Unified Query Work?

- Unify Metadata
  - Catalog data sources
  - Translate queries into plans
- Distribute Execution
  - Distribute the plan
  - Do work
  - Return answer

# Unifying Metadata

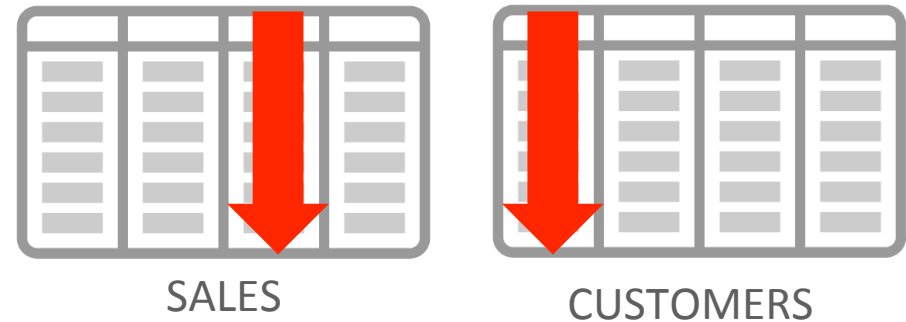
# Why Unify Metadata?

```
CREATE TABLE customers...  
SELECT name FROM customers
```

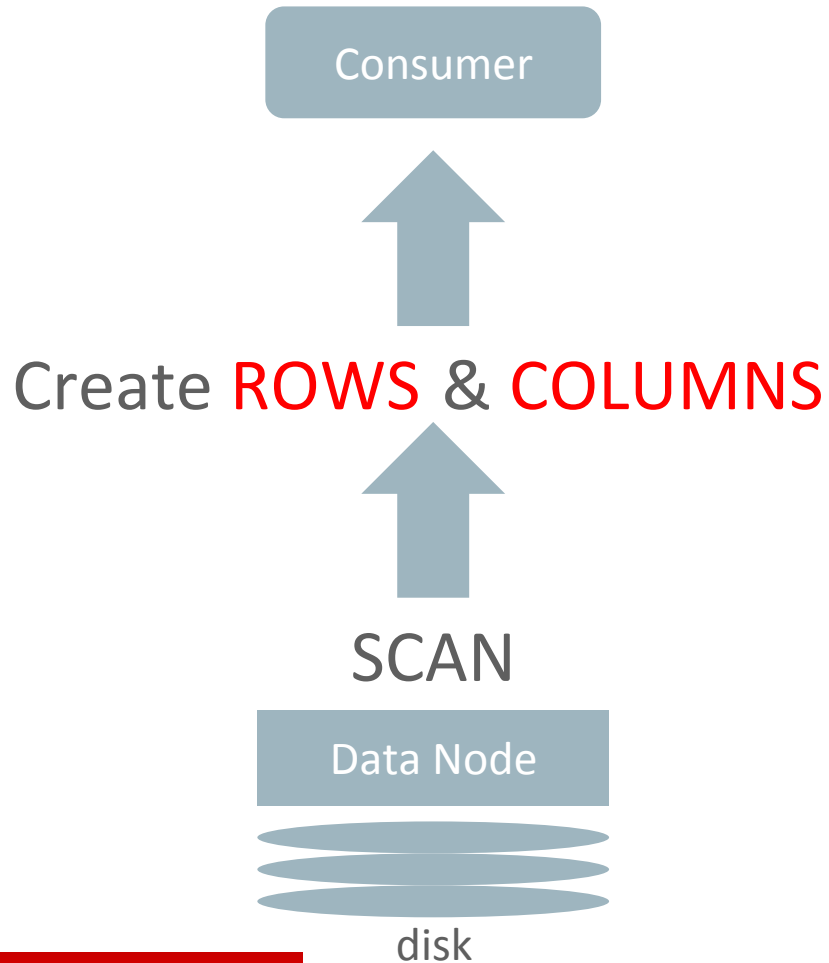


Query across sources → Integrate new metadata

```
SELECT customers.name, sales.amount
```



# Metadata: InputFormats and SerDes



- Scan and row creation needs to be able to work on “any” data format
- Data definitions and column deserializations are needed to provide a table

**RecordReader** => Scans data (keys and values)

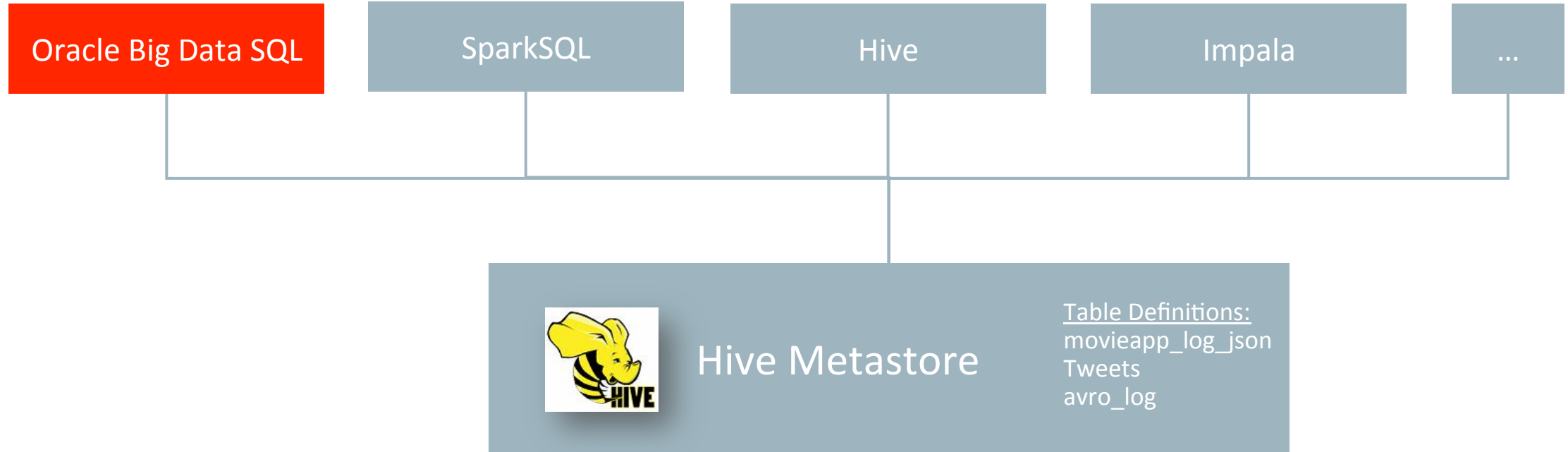
**InputFormat** => Defines parallelism

**SerDe** => Makes columns

**Metastore** => Maps DDL to Java access classes

# SQL-on-Hadoop Engines Share Metadata, not MapReduce

## Hive Metastore



Metastore maps DDL to Java access classes

# Extend Oracle External Tables

```
CREATE TABLE movielog (  
  click VARCHAR2(4000))  
ORGANIZATION EXTERNAL (  
  TYPE ORACLE_HIVE  
  DEFAULT DIRECTORY DEFAULT_DIR  
  ACCESS PARAMETERS  
  (  
    com.oracle.bigdata.tablename logs  
    com.oracle.bigdata.cluster mycluster  
  ))  
REJECT LIMIT UNLIMITED;
```

- New types of external tables
  - ORACLE\_HIVE (inherit metadata)
  - ORACLE\_HDFS (specify metadata)
- Access parameters for Big Data
  - Hadoop cluster
  - Remote Hive database/table
    - DBMS\_HADOOP Package for automatic import

# Enhance Oracle External Tables

```
CREATE TABLE ORDER (  
  cust_num VARCHAR2(10),  
  order_num VARCHAR2(20),  
  order_total NUMBER(8,2))  
ORGANIZATION EXTERNAL (  
  TYPE ORACLE_HIVE  
  DEFAULT DIRECTORY DEFAULT_DIR  
)  
PARALLEL 20  
REJECT LIMIT UNLIMITED;
```

- Transparent schema-for-read
  - Use fast C-based readers when possible
  - Use native Hadoop classes otherwise
- Engineered to understand parallelism
  - Map external units of parallelism to Oracle
- Architected for extensibility
  - StorageHandler capability enables future support for other data sources
  - Examples: MongoDB, HBase, Oracle NoSQL DB

That just makes a good **client.**



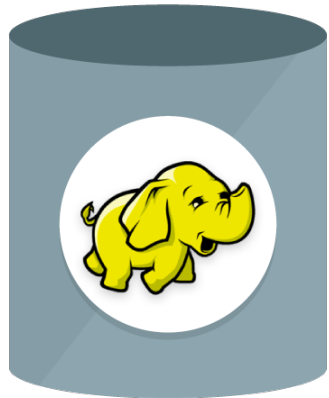
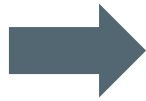
# Distribute Execution

# But how?

# Language-level Federation Fails

Been there, done that.

Hadoop Part



```
with sites as  
(  
  select s.custid as cust_id,  
  listagg(s.site, ',')  
    within group (order by  
  s.custid) site_list  
  from shortcodes s  
  group by custid
```

```
),  
select c.first_name,  
c.last_name, c.AGE,  
c.state_province, s.site_list  
from customers c, sites s;
```



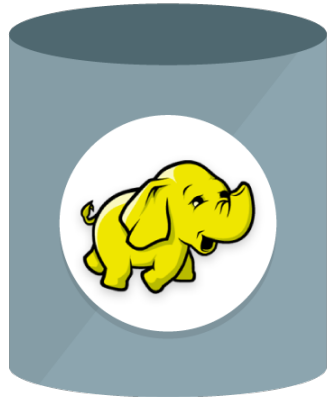
Database Part



# Language-level Federation Fails

Been there, done that.

```
select s.custid as cust_id,  
       listagg(s.site, ',')  
         within group (order by  
                       s.custid) site_list  
from short_codes  
group by custid
```



- Operators exist in both places?
- Is their behavior consistent?
- How do you negotiate resources?

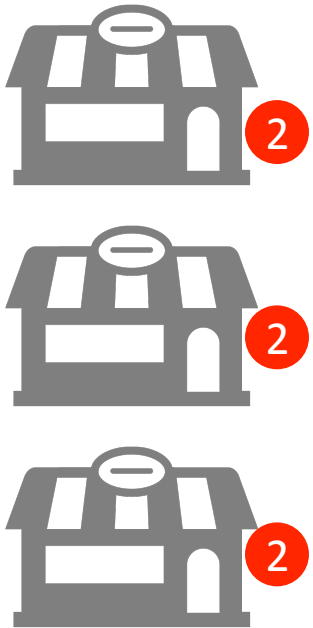


We have to do **better**

**Query Franchising** – *dispatch of query processing to self-similar compute agents on disparate systems without loss of operational fidelity*

What does **that** mean?

# Query Franchising: Uniform Behavior, Disparate Location



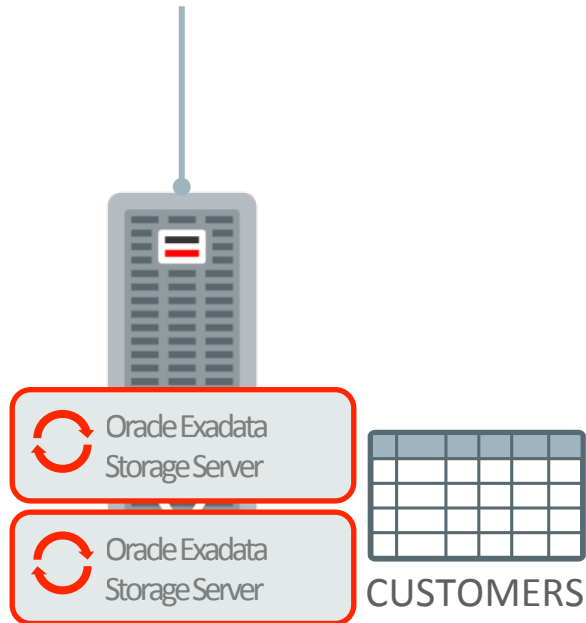
- 1** Top-level plan created
  - Holistic plan plan for all work
  - Distribute to franchises based by location
- 2** Franchisees carry out local work
  - Franchises secure and utilize resources
  - All franchises speak the internal language
- 3** Global operations optimized
  - Adapts to local variation
  - Nothing “lost in translation”



# What Can Big Data Learn from Exadata?

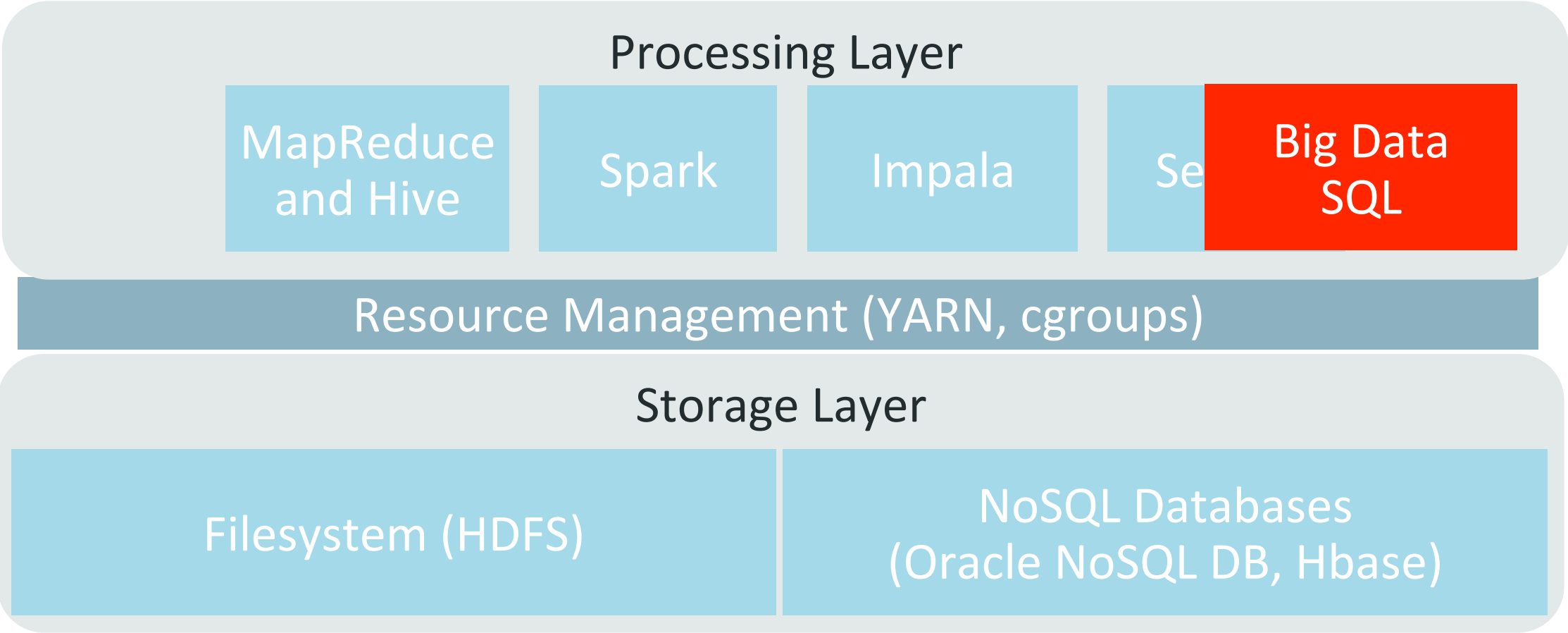
## Query Federation for Oracle Database

```
SELECT name, SUM(purchase)
FROM customers
GROUP BY name;
```

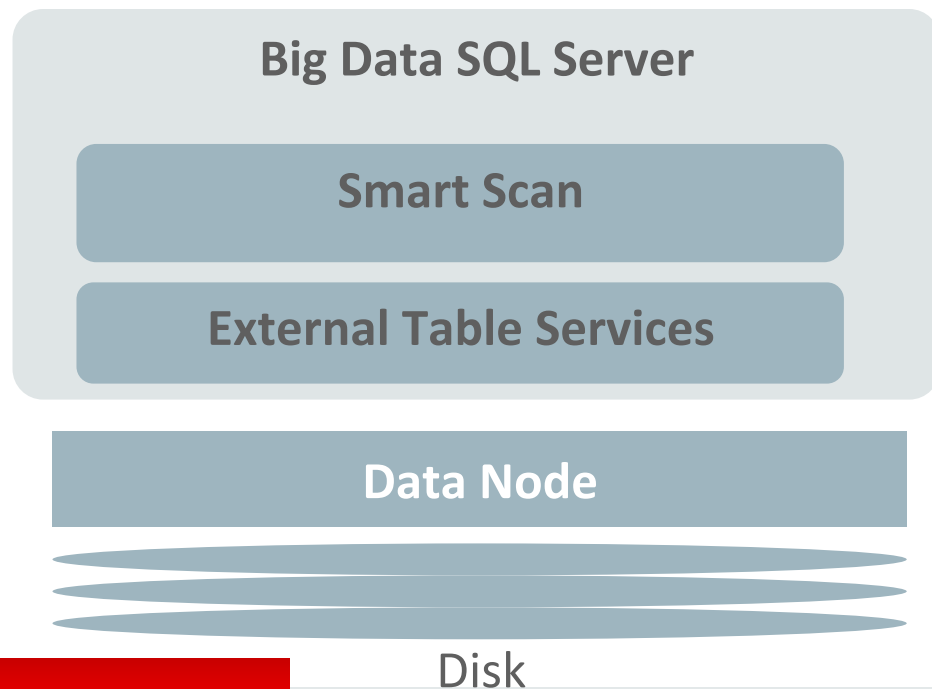


- 1 Oracle SQL query issued
  - Plan constructed
  - Query executed
- 2 Smart Scan Works on Storage
  - Filter out unneeded rows
  - Project only queried columns
  - Score data models
  - Bloom filters to speed up joins

# Big Data SQL Server: A New Hadoop Processing Engine



# Smart Scan for Hadoop: Optimizing Performance



## “Oracle on top”

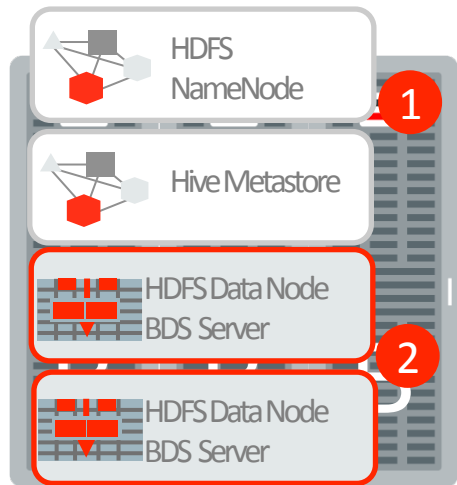
- Apply filter predicates
- Project columns
- Parse semi-structured data

## “Hadoop on the bottom”

- Work close to the data
- Schema-on-read with Hadoop classes
- Transformation into Oracle data stream

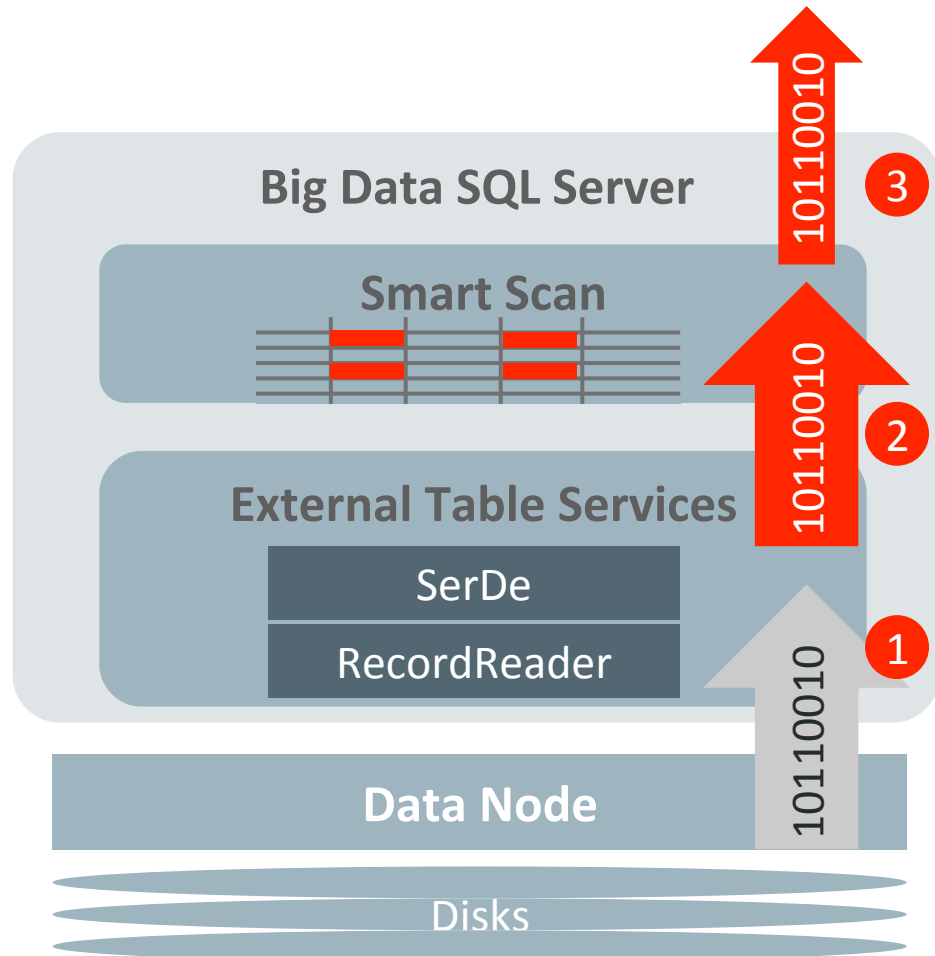
# Big Data SQL Query Execution

## How do we query Hadoop?



- 1** Query compilation determines:
  - Data locations
  - Data structure
  - Parallelism
- 2** Fast reads using Big Data SQL Server
  - Schema-for-read using Hadoop classes
  - Smart Scan selects only relevant data
- 3** Process filtered result
  - Move relevant data to database
  - Join with database tables
  - Apply database security policies

# Big Data SQL Dataflow



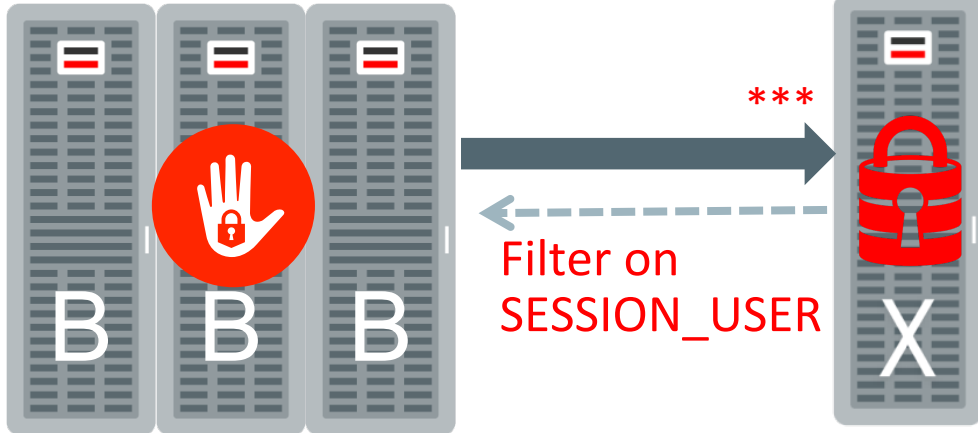
- 1** Read data from HDFS Data Node
  - Direct-path reads
  - C-based readers when possible
  - Use native Hadoop classes otherwise
- 2** Translate bytes to Oracle
- 3** Apply Smart Scan to Oracle bytes
  - Apply filters
  - Project Columns
  - Parse JSON/XML
  - Score models

# But How Does Security Work?

```

DBMS_REDACT.ADD_POLICY(
  object_schema => 'MCLICK',
  object_name => 'TWEET_V',
  column_name => 'USERNAME',
  policy_name => 'tweet_redaction',
  SYS_CONTEXT('USERENV','SESSION_USER');
  function_type => DBMS_REDACT.PARTIAL,
  function_parameters =>
  'VVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVVV,* ,3,25',
  expression => '1=1'
);

```



- 1 Database security for query access
  - Virtual Private Databases
  - Redaction
  - Audit Vault and Database Firewall
- 2 Hadoop security for Hadoop jobs
  - Kerberos Authentication
  - Apache Sentry (RBAC)
  - Audit Vault
- 3 System-specific encryption
  - Database tablespace encryption
  - BDA On-disk Encryption

# Customer 360, **Live!**

# Unified Query Means **Less** Lock-in

- Unified Query means
  - More innovation
  - Less risk
- Use the right tools for your job
  - Hadoop, NoSQL, and whatever's next
  - We'll query all of it
- Explore and adopt new technologies
  - Focus on creating value using your data
  - We'll build bridges back to the business



# **Hardware and Software Engineered to Work Together**

ORACLE®