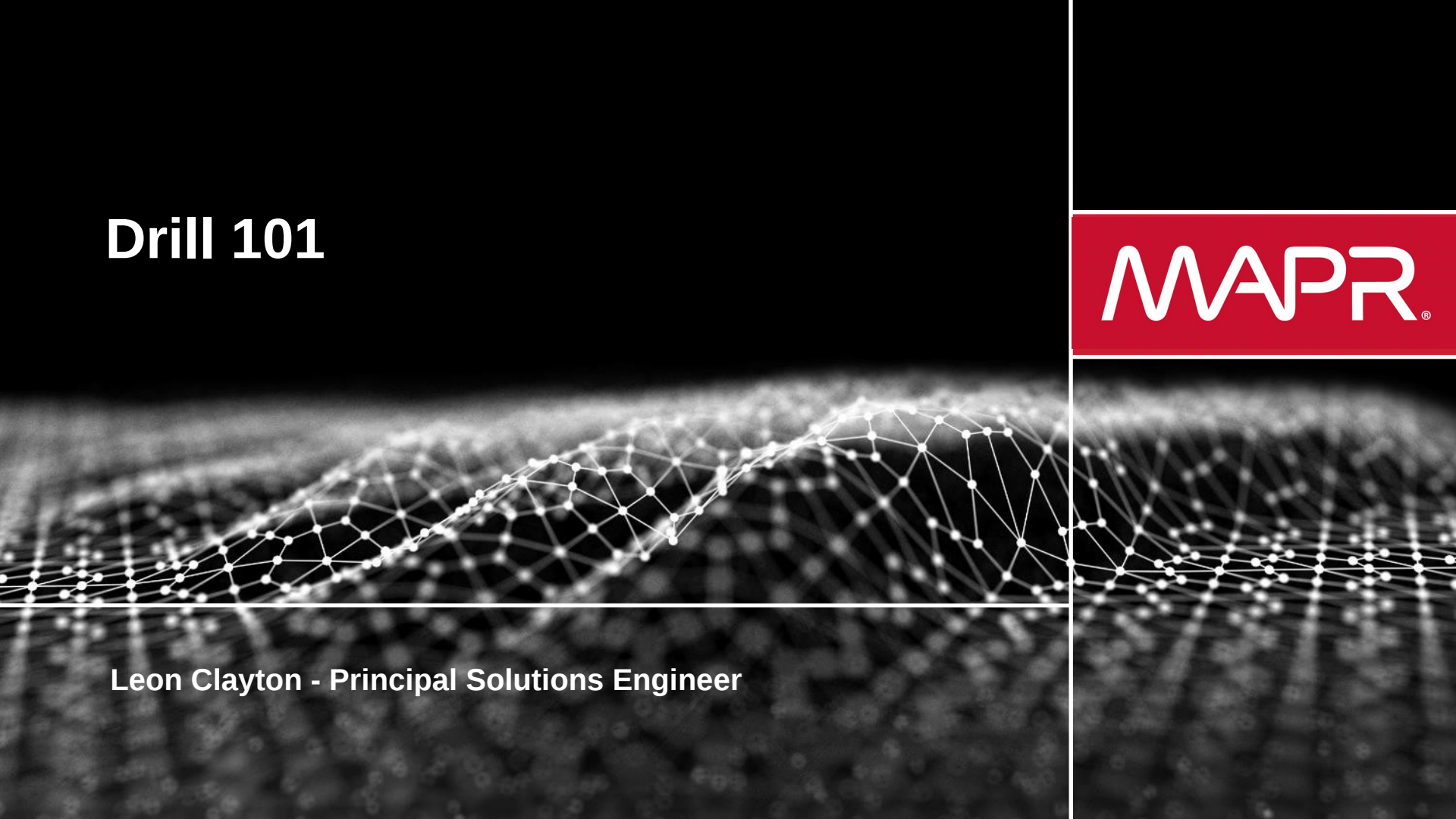
Drill 101



Leon Clayton - Principal Solutions Engineer

{"about" : "me"}

Leon Clayton

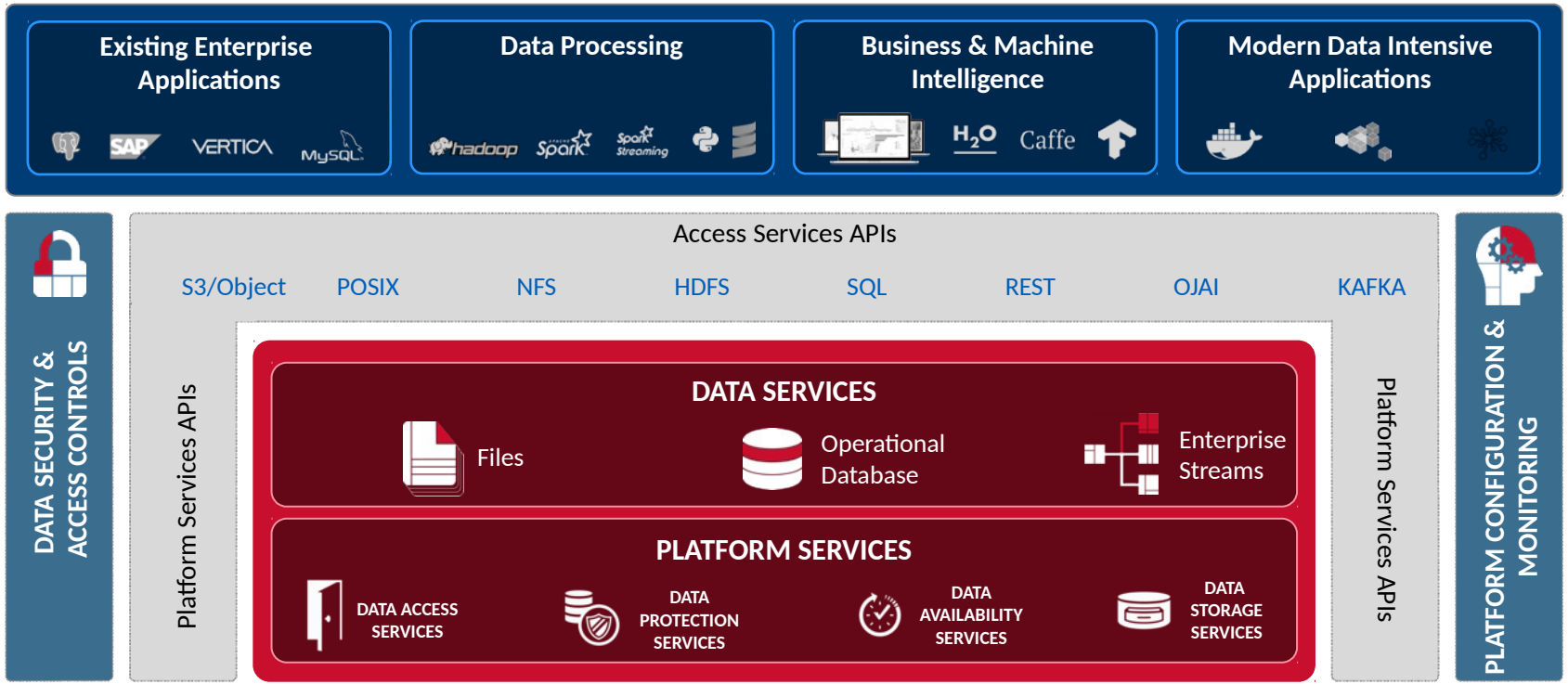
- MapR
  - Solution Architect
- EMC
  - EMEA Isilon Lead
- NetApp
  - Performance Team Lead
- Philips
  - HPC team lead
- Royal Navy
  - Type 42 – Real time systems engineer

- lclayton@mapr.com

MAPR

Foundation For A Global Data Fabric

MapR Converged Data Platform



Agenda

- Why Drill?
- Some Drilling with Open Data
- How does it work?

Data Increasingly Stored in *Non-Relational* Datastores

Volume	GBs-TBs	TBs-PBs
Structure	Structured	Structured, semi-structured and unstructured
Development	Planned (release cycle = months-years)	Iterative (release cycle = days-weeks)
Database	RELATIONAL DATABASES ORACLE, Microsoft SQL Server, MySQL **Fixed schema** DBA controls structure	NON-RELATIONAL DATASTORES hadoop, APACHE HBASE, mongoDB **Dynamic / Flexible schema** Application controls structure

1980 1990 2000 2010 2020

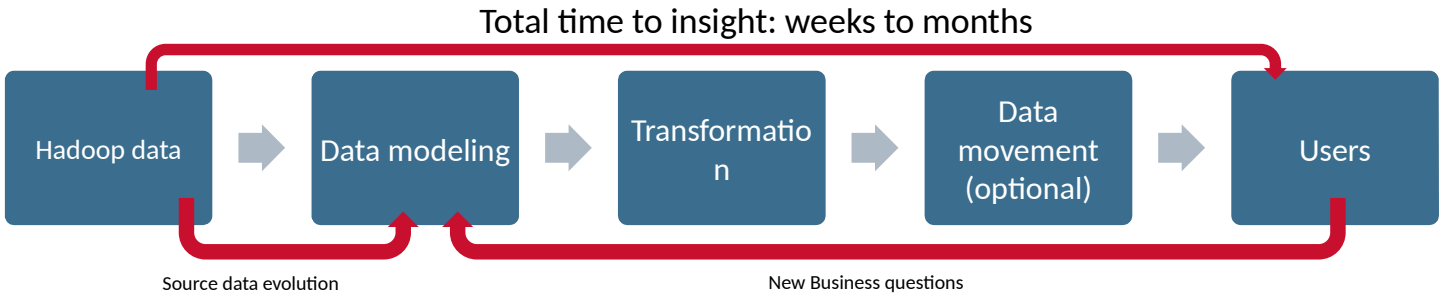
MAPR

How To Bring SQL to Non-Relational Data Stores?

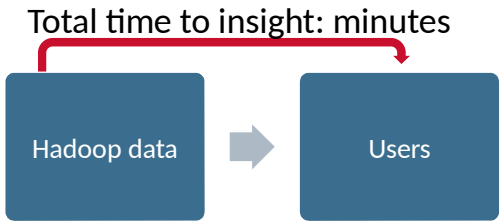
Familiarity of SQL	Agility of NoSQL
• ANSI SQL semantics • Low latency • Integrated with Tools/Applications 	• No schema management – HDFS (Parquet, JSON, etc.) – HBase – … • No transformation – No silos of data • Ease of use

Enabling “As-It-Happens” Business with Instant Analytics

Traditional approach



Exploratory approach



Drill Supports *Schema Discovery On-The-Fly*

Schema Declared In Advance	Schema Discovered On-The-Fly
• Fixed schema • Leverage schema in centralized repository (Hive Metastore)	• Fixed schema, evolving schema or schema-less • Leverage schema in centralized repository or self-describing data

SCHEMA ON WRITE

SCHEMA BEFORE READ

SCHEMA ON THE FLY

Contributing to Apache Drill

- Pioneering Data Agility for Hadoop
- Apache open source project
- Scale-out execution engine for low-latency queries
- Unified SQL-based API for analytics & operational applications

150+ years of experience building databases and distributed systems

ORACLE

IBM DB2

Microsoft SQL Server

VERTICA An HP Company

ParAccel

50+ contributors

LinkedIn

WISCONSIN UNIVERSITY OF WISCONSIN–MADISON

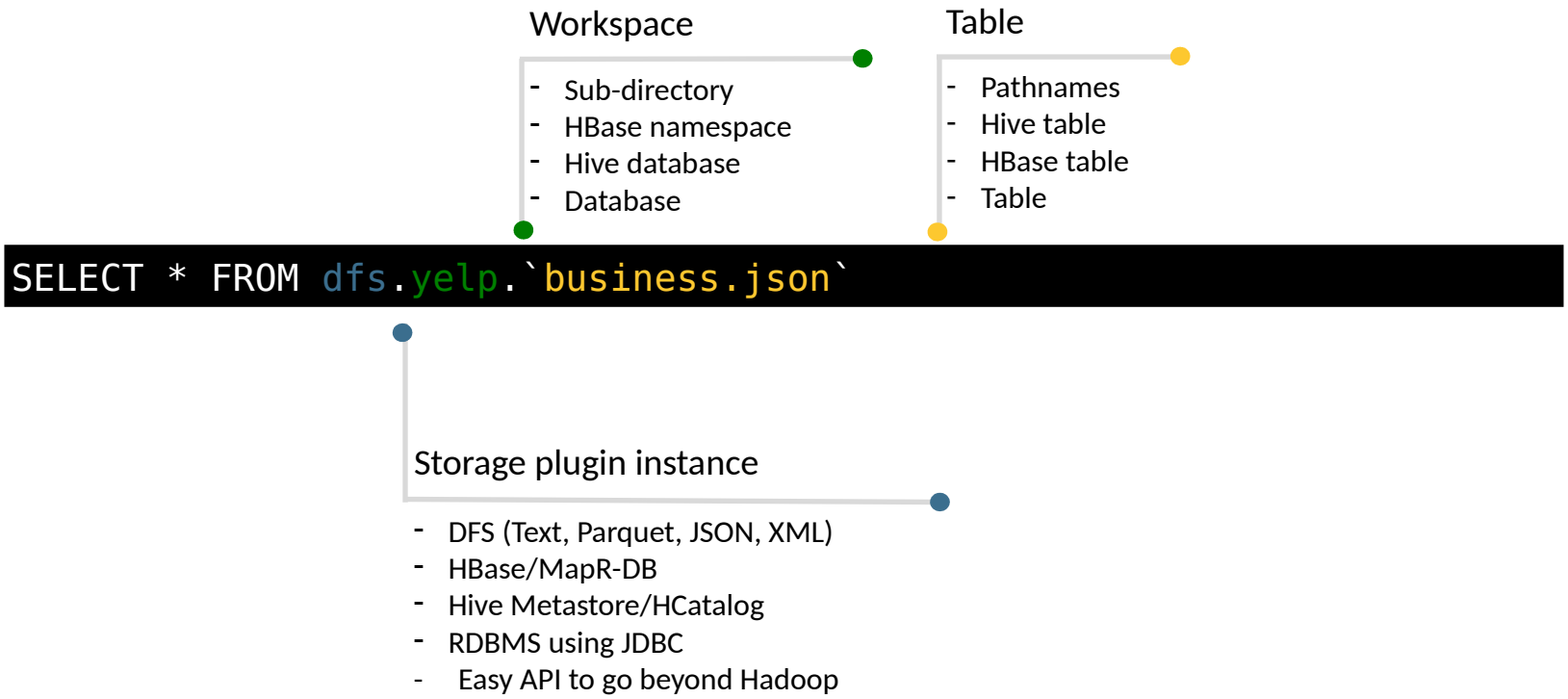
dremio

pentaho

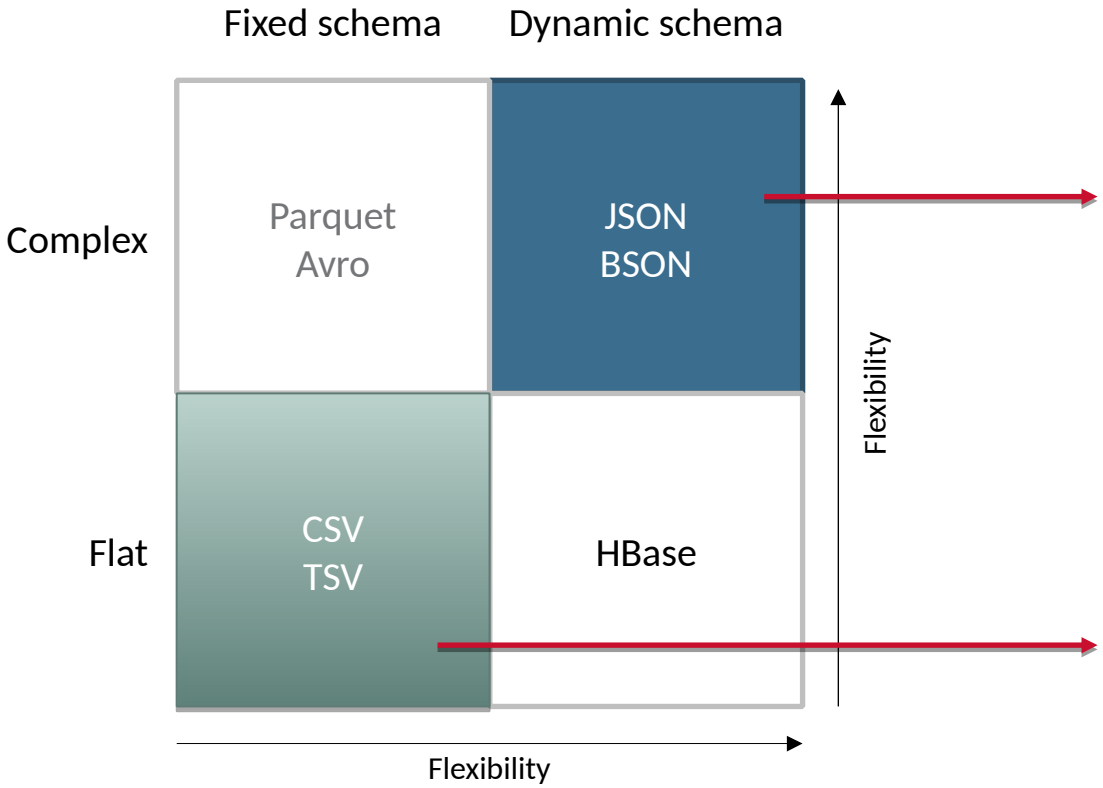
CISCO

MAPR

Drill Enables ‘SQL-on-Everything’



Drill's Data Model is Flexible



Apache Drill table

```
{
  name: {
    first: Michael,
    last: Smith
  },
  hobbies: [ski, soccer],
  district: Los Altos
}
{
  name: {
    first: Jennifer,
    last: Gates
  },
  hobbies: [sing],
  preschool: CCLC
}
```

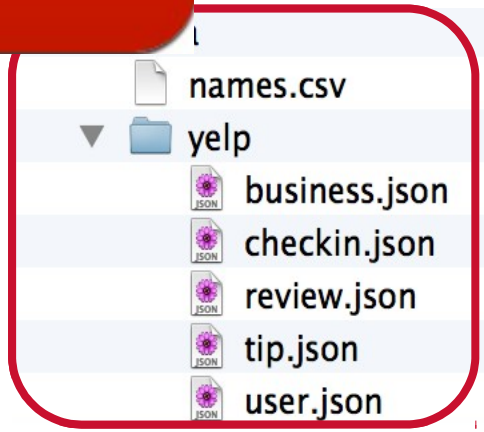
RDBMS/SQL-on-Hadoop table

Name	Gender	Age
Michael	M	6
Jennifer	F	3

Drill into Data

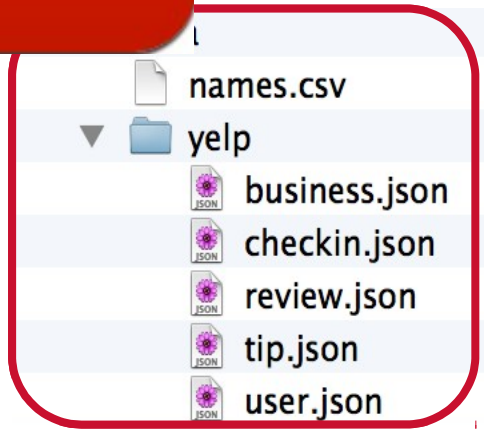
http://www.yelp.com/dataset_challenge

s dataset




```
{
        "business_id": "4bEjOyTaDG24SY5TxsaUNQ",
        "full_address": "3655 Las Vegas Blvd S\nThe Strip\nLas Vegas, NV 89109",
        "hours": {
                "Monday": {"close": "23:00", "open": "07:00"},
                "Tuesday": {"close": "23:00", "open": "07:00"},
                "Friday": {"close": "00:00", "open": "07:00"},
                "Wednesday": {"close": "23:00", "open": "07:00"},
                "Thursday": {"close": "23:00", "open": "07:00"},
                "Sunday": {"close": "23:00", "open": "07:00"},
                "Saturday": {"close": "00:00", "open": "07:00"}
        },
        "open": true,
        "categories": ["Breakfast & Brunch", "Steakhouses", "French", "Restaurants"],
        "city": "Las Vegas",
        "review_count": 4084,
        "name": "Mon Ami Gabi",
        "neighborhoods": ["The Strip"],
        "longitude": -115.172588519464,
        "state": "NV",
        "stars": 4.0,
        "attributes": {
                "Alcohol": "full_bar",
                "Noise Level": "average",
                "Has TV": false,
                "Attire": "casual",
                "Ambience": {
                        "romantic": true,
                        "intimate": false,
                        "touristy": false,
                        "hipster": false,
                        "classy": true,
                        "trendy": false,
                        "casual": false
                },
                "Good For": {"dessert": false, "latenight": false, "lunch": false,
                        "dinner": true, "breakfast": false, "brunch": false},
        }
}
```

dataset




```
{
  "votes": {"funny": 0, "useful": 2, "cool": 1},
  "user_id": "Xqd0DzHaiyRqVH3WRG7hzg",
  "review_id": "15SdjuK7DmYqUAj6rjGowg",
  "stars": 5,
  "date": "2007-05-17",
  "text": "dr. goldberg offers everything ...",
  "type": "review",
  "business_id": "vcNAWiLM4dR7D2nwwJ7nCA"
}
```

```
$ tar -xvzf apache-drill-1.3.0.tar.gz

$ bin/sqlline -u jdbc:drill:zk=local
$ bin/drill-embedded

> SELECT state, city, count(*) AS businesses
  FROM dfs.yelp.`business.json.gz`
  GROUP BY state, city
  ORDER BY businesses DESC LIMIT 10;
```

state	city	businesses
NV	Las Vegas	12021
AZ	Phoenix	7499
AZ	Scottsdale	3605
EDH	Edinburgh	2804
AZ	Mesa	2041
AZ	Tempe	2025
NV	Henderson	1914
AZ	Chandler	1637
WI	Madison	1630
AZ	Glendale	1196

Intuitive SQL Access to Complex Data

```
// It's Friday 10pm in Vegas and looking for Hummus

> SELECT name, stars, b.hours.Friday friday, categories
  FROM dfs.yelp.`business.json` b
  WHERE b.hours.Friday.`open` < '22:00' AND
        b.hours.Friday.`close` > '22:00' AND
        REPEATED_CONTAINS(categories, 'Mediterranean') AND
        city = 'Las Vegas'
  ORDER BY stars DESC
  LIMIT 2;
```

name	stars	friday	categories
Khoury's Mediterranean Restaurant	4.0	{"close":"23:00","open":"11:00"}	["Greek","Mediterranean","Middle Eastern","Restaurants"]
Olives	4.0	{"close":"22:30","open":"11:00"}	["Bars","Mediterranean","Nightlife","Restaurants"]

ANSI SQL Compatibility

```
//Get top cool rated businesses

SELECT b.name from dfs.yelp.`business.json` b
  WHERE b.business_id IN
 (SELECT r.business_id FROM dfs.yelp.`review.json` r
  GROUP BY r.business_id HAVING SUM(r.votes.cool) > 2000 ORDER BY
  SUM(r.votes.cool) DESC);

+---------------------------------+
|              name               |
+---------------------------------+
| Earl of Sandwich                |
| XS Nightclub                    |
| The Cosmopolitan of Las Vegas   |
| Wicked Spoon                    |
| Bacchanal Buffet                |
+---------------------------------+
```

Logical Views

```
//Create a view combining business and reviews datasets

> CREATE OR REPLACE VIEW dfs.tmp.BusinessReviews AS
    SELECT b.name, b.stars, r.votes.funny,
           r.votes.useful, r.votes.cool, r.`date`
      FROM dfs.yelp.`business.json` b, dfs.yelp.`review.json` r
      WHERE r.business_id = b.business_id;
+-------+-----------------------------------------------------------+
|  ok   |                          summary                          |
+-------+-----------------------------------------------------------+
| true  | View 'BusinessReviews' replaced successfully in 'dfs.tmp' schema  |
+-------+-----------------------------------------------------------+

> SELECT COUNT(*) AS Total FROM dfs.tmp.BusinessReviews;

+------------+
|    Total   |
+------------+
|  1125458   |
+------------+
```

Materialized Views AKA Tables

```
> ALTER SESSION SET `store.format` = 'parquet';

> CREATE TABLE dfs.yelp.BusinessReviewsTbl AS
    SELECT b.name, b.stars, r.votes.funny funny,
           r.votes.useful useful, r.votes.cool cool, r.`date`
      FROM dfs.yelp.`business.json` b, dfs.yelp.`review.json` r
      WHERE r.business_id = b.business_id;

+------------+--------------------------+
|  Fragment  | Number of records written |
+------------+--------------------------+
| 1_0        | 176448                   |
| 1_1        | 192439                   |
| 1_2        | 198625                   |
| 1_3        | 200863                   |
| 1_4        | 181420                   |
| 1_5        | 175663                   |
+------------+--------------------------+
```

Repeated Values Support

```
// Flatten repeated categories

> SELECT name, categories
  FROM dfs.yelp.`business.json` LIMIT 3;

+-------------------------+-----------------------------------+
|          name           |            categories             |
+-------------------------+-----------------------------------+
| Eric Goldberg, MD       | ["Doctors","Health & Medical"]    |
| Clancy's Pub            | ["Nightlife"]                     |
| Cool Springs Golf Center| ["Active Life","Mini Golf","Golf"]|
+-------------------------+-----------------------------------+

> SELECT name, FLATTEN(categories) AS categories
  FROM dfs.yelp.`business.json` LIMIT 5;
+-------------------------+------------------+
|          name           |    categories    |
+-------------------------+------------------+
| Eric Goldberg, MD       | Doctors          |
| Eric Goldberg, MD       | Health & Medical |
| Clancy's Pub            | Nightlife        |
| Cool Springs Golf Center| Active Life      |
| Cool Springs Golf Center| Mini Golf        |
+-------------------------+------------------+
```

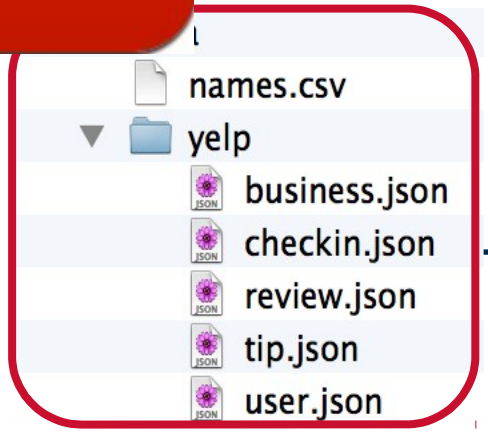
Extensions to ANSI SQL to work with repeated values

```
// Get most common business categories

>SELECT category, count(*) AS categorycount
  FROM (SELECT name, FLATTEN(categories) AS category
        FROM dfs.yelp.`business.json`) c
  GROUP BY category ORDER BY categorycount DESC;

+-------------------------------+----------------+
|           category            | categorycount  |
+-------------------------------+----------------+
| Restaurants                   | 21892          |
| Shopping                      | 8919           |
| Automotive                    | 2965           |
… … … …… … … …… … … …… … … …… … … …… … … …… … …
| Oriental                      | 1              |
| High Fidelity Audio Equipment | 1              |
+-------------------------------+----------------+
```

Checkins dataset




```
{
   "checkin_info":{
      "3-4":1,
      "13-5":1,
      "6-6":1,
      "14-5":1,
      "14-6":1,
      "14-2":1,
      "14-3":1,
      "19-0":1,
      "11-5":1,
      "13-2":1,
      "11-6":2,
      "11-3":1,
      "12-6":1,
      "6-5":1,
      "5-5":1,
      "9-2":1,
      "9-5":1,
      "9-6":1,
      "5-2":1,
      "7-6":1,
      "7-5":1,
      "7-4":1,
      "17-5":1,
      "8-5":1,
      "10-2":1,
      "10-5":1,
      "10-6":1
   },
   "type":"checkin",
   "business_id":"JwUE5GmEO-sH1FuwJgKBlQ"
}
```

Supports Dynamic / Unknown Columns

```
> SELECT KVGEN(checkin_info) checkins
   FROM dfs.yelp.`checkin.json` LIMIT 1;
+------------+
|  checkins  |
+------------+
| [{"key":"3-4","value":1},{"key":"13-5","value":1},{"key":"6-6","value":1},{"key":"14-
5","value":1},{"key":"14-6","value":1},{"key":"14-2","value":1},{"key":"14-3","value":1},{"key":"19-
0","value":1},{"key":"11-5","value":1},{"key":"13-2","value":1},{"key":"11-6","value":2},{"key":"11-
3","value":1},{"key":"12-6","value":1},{"key":"6-5","value":1},{"key":"5-5","value":1},{"key":"9-
2","value":1},{"key":"9-5","value":1},{"key":"9-6","value":1},{"key":"5-2","value":1},{"key":"7-
6","value":1},{"key":"7-5","value":1},{"key":"7-4","value":1},{"key":"17-5","value":1},{"key":"8-
5","value":1},{"key":"10-2","value":1},{"key":"10-5","value":1},{"key":"10-6","value":1}] |
+------------+

> SELECT FLATTEN(KVGEN(checkin_info)) checkins FROM
dfs.yelp.`checkin.json` limit 6;

+--------------------------+
|         checkins         |
+--------------------------+
| {"key":"9-5","value":1}  |
| {"key":"7-5","value":1}  |
| {"key":"13-3","value":1} |
| {"key":"17-6","value":1} |
| {"key":"13-0","value":1} |
| {"key":"17-3","value":1} |
+--------------------------+
```

Makes it easy to work with dynamic/unknown columns

```
// Count total number of checkins on Sunday midnight

> SELECT SUM(checkintbl.checkins.`value`) as
SundayMidnightCheckins FROM
   (SELECT FLATTEN(KVGEN(checkin_info)) checkins
    FROM dfs.yelp.checkin.json`) checkintbl
    WHERE checkintbl.checkins.key='23-0';

+-----------------------+
| SundayMidnightCheckins |
+-----------------------+
| 8575                  |
+-----------------------+
```

Federated Queries

```
// Join JSON File, Parquet and MongoDB collection

> SELECT u.name, b.category,  count(1) nb_review
FROM mongo.yelp.`user` u , dfs.yelp.`review.parquet` r,  (select business_id, flatten(categories)
category from dfs.yelp.`business.json` ) b
WHERE u.user_id = r.user_id
AND b.business_id = r.business_id
GROUP BY u.user_id, u.name, b.category
ORDER BY nb_review DESC
LIMIT 10;

+-----------+--------------+------------+
|   name    |   category   | nb_review  |
+-----------+--------------+------------+
| Rand      | Restaurants  | 1086       |
| J         | Restaurants  | 661        |
| Jennifer  | Restaurants  | 657        |
| Brad      | Restaurants  | 638        |
| Jade      | Restaurants  | 586        |
… … … … … … … … … .. .. ..
| Emily     | Restaurants  | 560        |
| Norm      | Restaurants  | 543        |
| Aileen    | Restaurants  | 499        |
| Michael   | Restaurants  | 496        |
+-----------+--------------+------------+
```

Drill Data Sources

Directories are implicit partitions

```
logs
├── 2014
│   ├── 1
│   ├── 2
│   ├── 3
│   └── 4
└── 2015
    └── 1
```

```
select dir0, count(1)
from dfs.logs.`*`
where dir1 in (1,2,3)
group by dir0
```

Hands-On

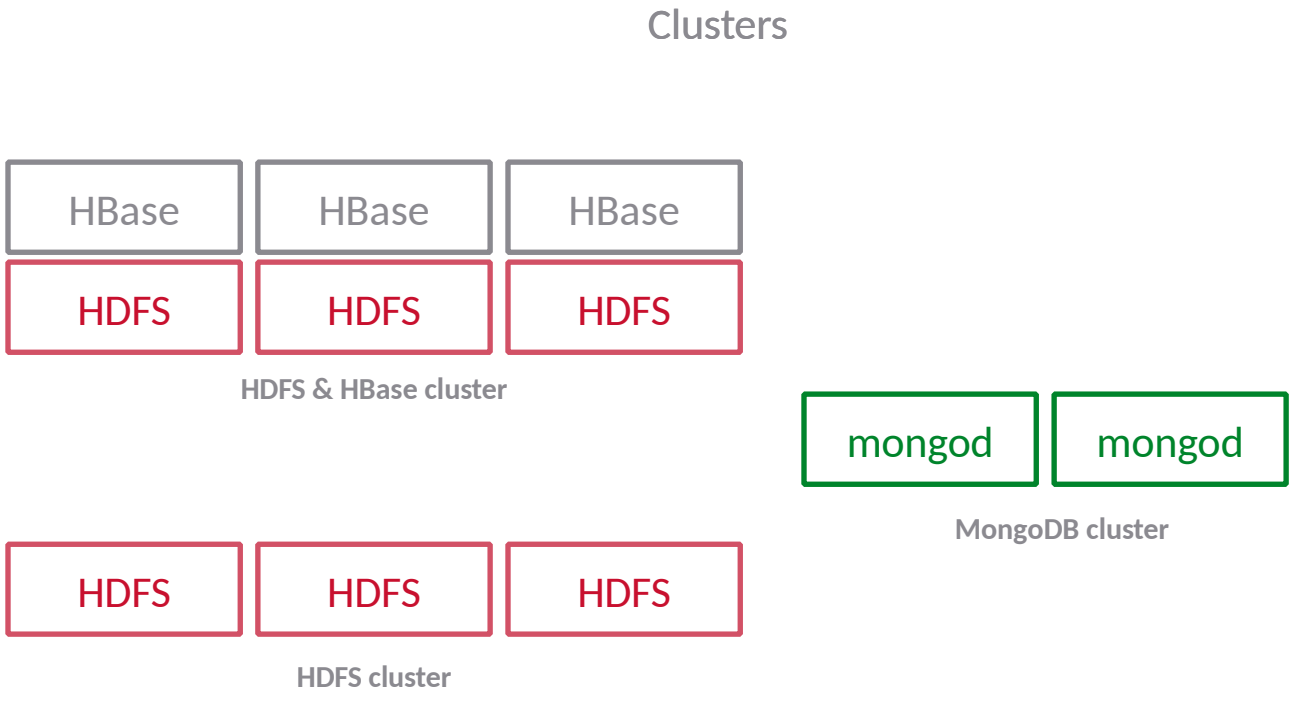
https://github.com/tgrall/drill-workshop

How does it work?

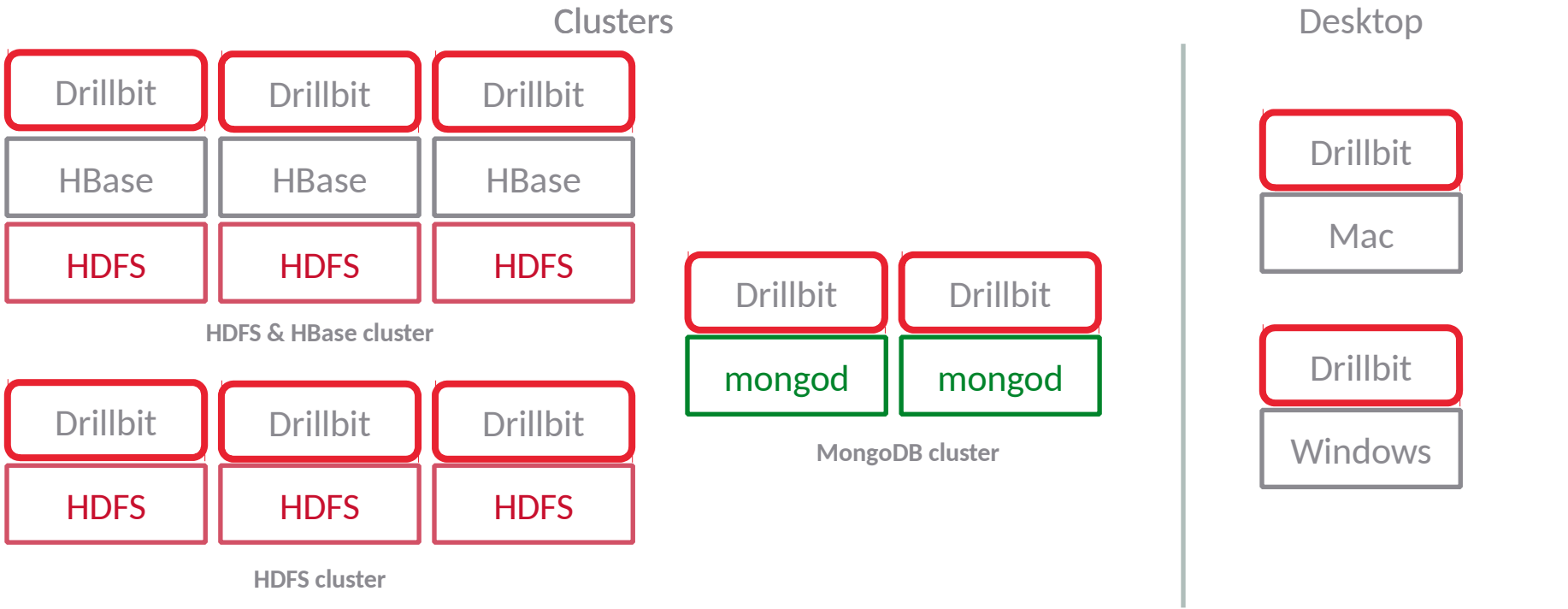
Everything is a “Drillbit”

- “Drillbits” run on each node
- In-memory columnar execution
- Coordination, Planning, Execution
- Networked or not
- Exposes JDBC, ODBC, REST
- Built In Web UI and CLI
- Extensible
  - Custom Functions
  - Data Sources

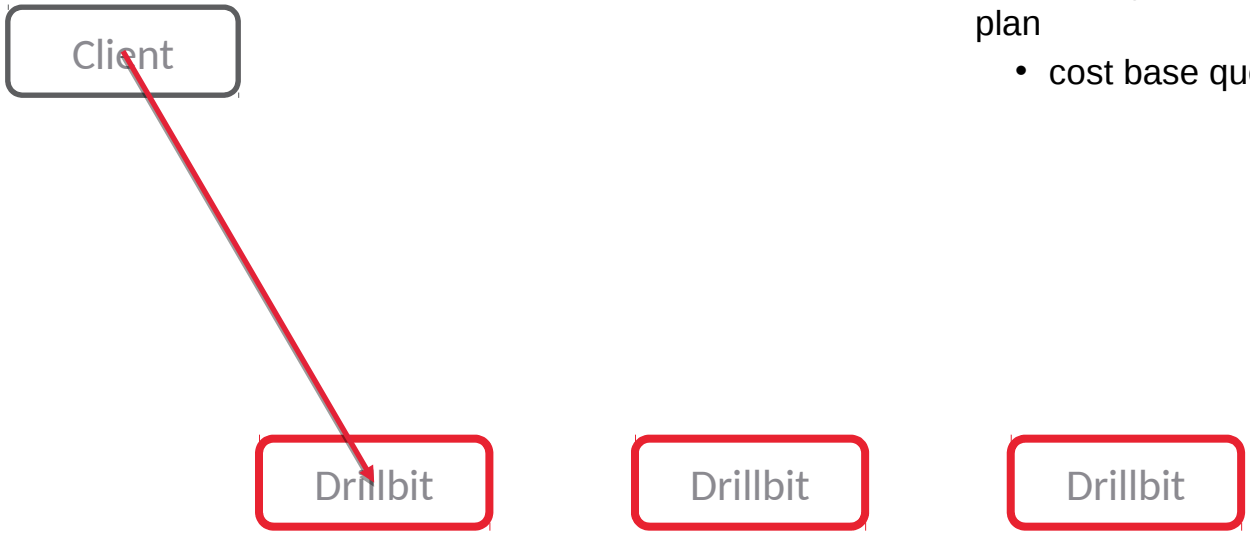
Data Locality



Run drillbits close to your data!



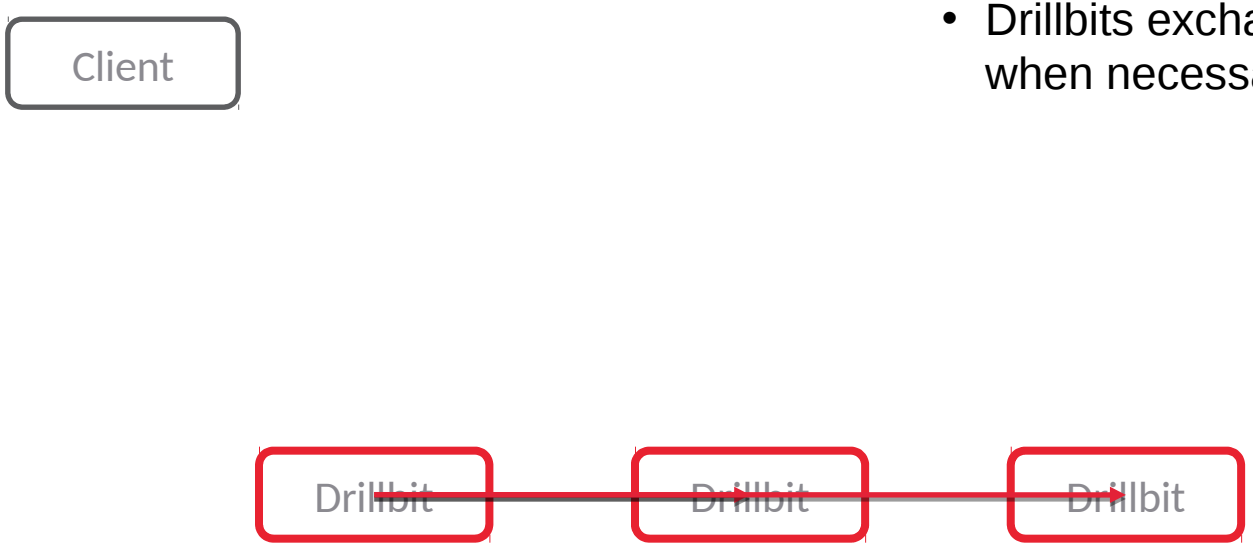
Query Execution



- Client connects to **“a” Drillbit**
- This Drillbit becomes the **foreman**
- Foreman generates execution plan
  - cost base query optimisation

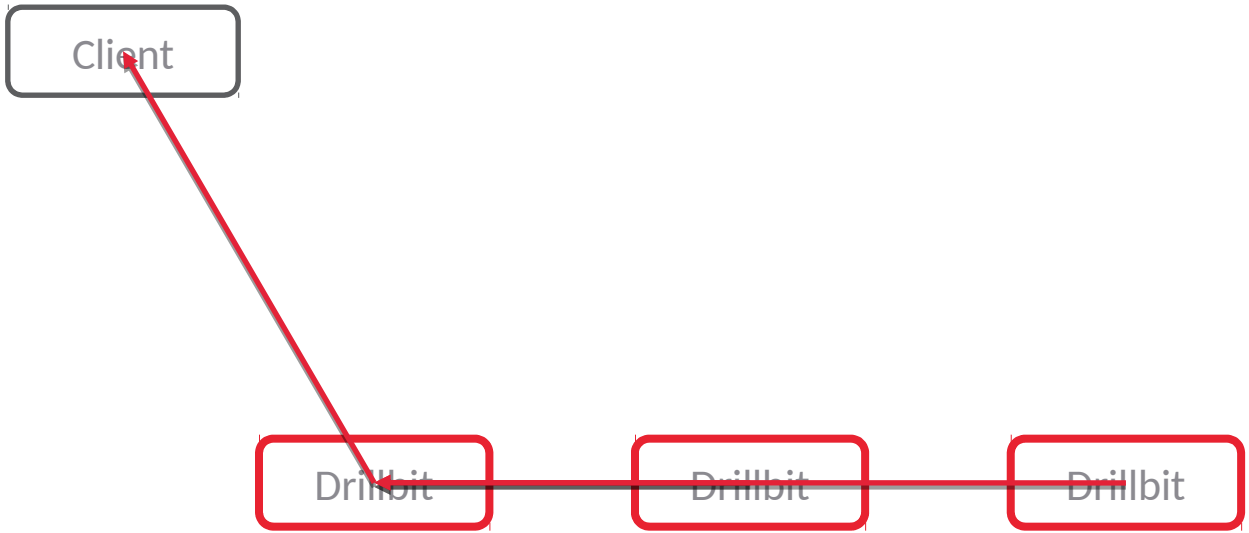
Query Execution

- Execution fragment are farmed to other Drillbits
- Drillbits exchange data when necessary



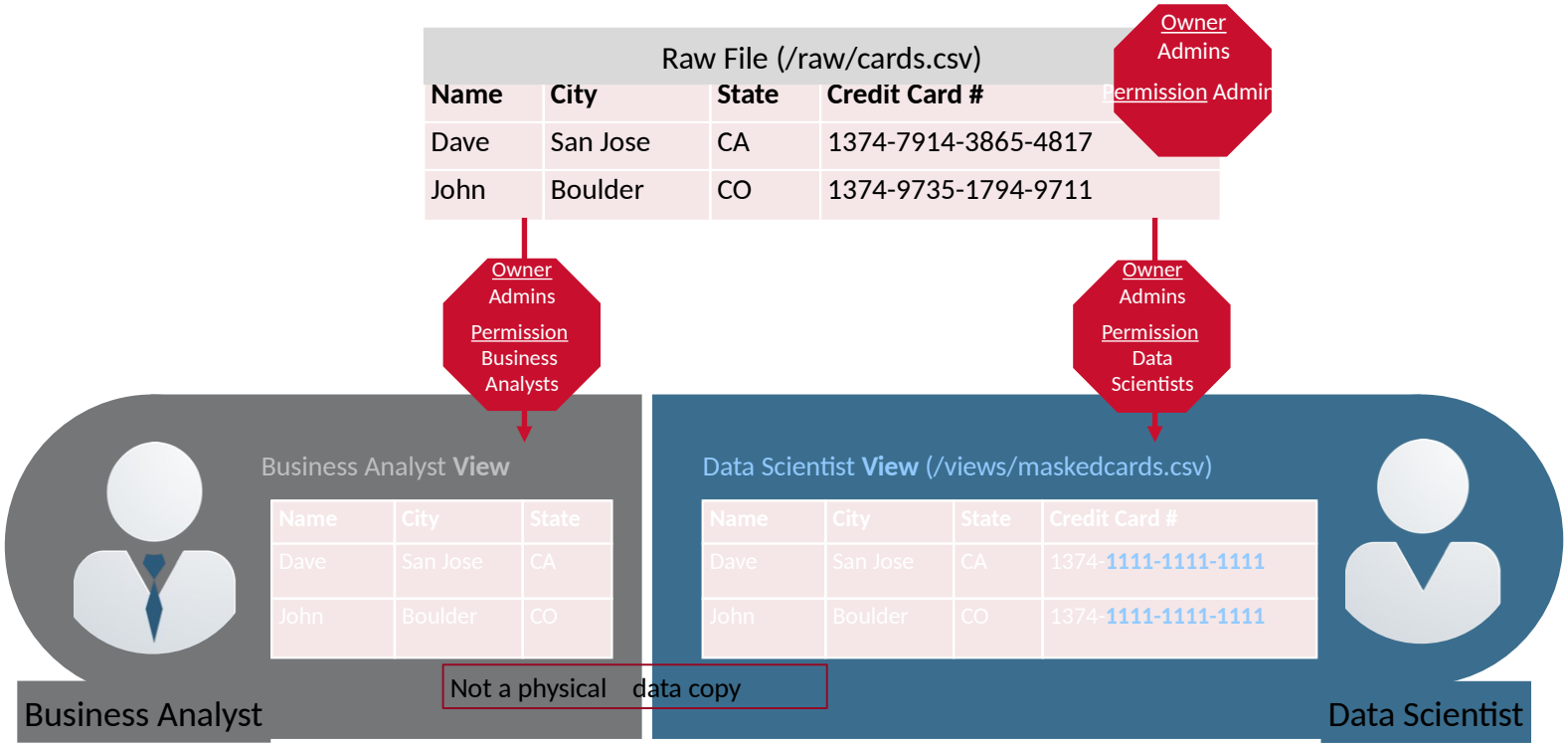
Query Execution

- Results are returned to the user through foreman



Security?

Granular Security via Drill Views



Extensibility

Extending Apache Drill

- File Format
  - ORC
  - …
- Data Sources
  - NoSQL Databases
  - Search Engines
  - REST
  - ….
- Custom Functions

https://github.com/mapr-demos/simple-drill-functions

Recommendations for Getting Started with Drill

New to Drill?

- Get started with Free MapR On Demand training
- Test Drive Drill on cloud with AWS
- Learn how to use Drill with Hadoop using MapR sandbox

Ready to play with your data?

- Try out Apache Drill in 10 mins guide on your desktop
- Download Drill for your cluster and start exploration
- Comprehensive tutorials and documentation available

Ask questions

- user@drill.apache.org
- dev@drill.apache.com