Massachusetts
Institute of
Technology

DBg Database Group
MIT Computer Science and Artificial Intelligence Lab

Aurum: A Large Scale Data Discovery System Based on a Source Retrieval Algebra

Raul Castro Fernandez,
Samuel Madden, Michael Stonebraker
MIT

January 27th
NEDB 2017

Employee gender distribution per department

Employee gender distribution per department

```
SELECT department, gender, COUNT(*)
FROM Employee
GROUP BY gender, department;
```

Employee Id	Name	Gender	Department
1001	John	Male	Finance
1002	Mary	Female	Tech
1003	Susan	Female	Finance

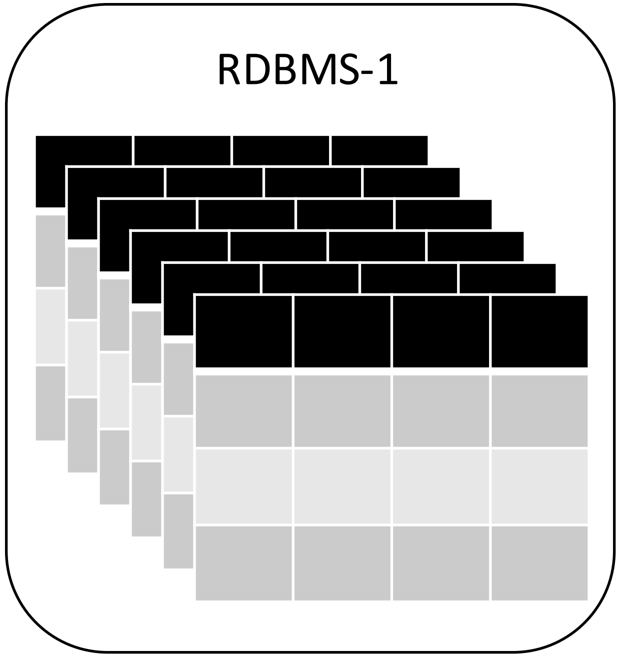
Employee gender distribution per department

```
SELECT department, gender, COUNT(*)
FROM Employee
GROUP BY gender, department;
```

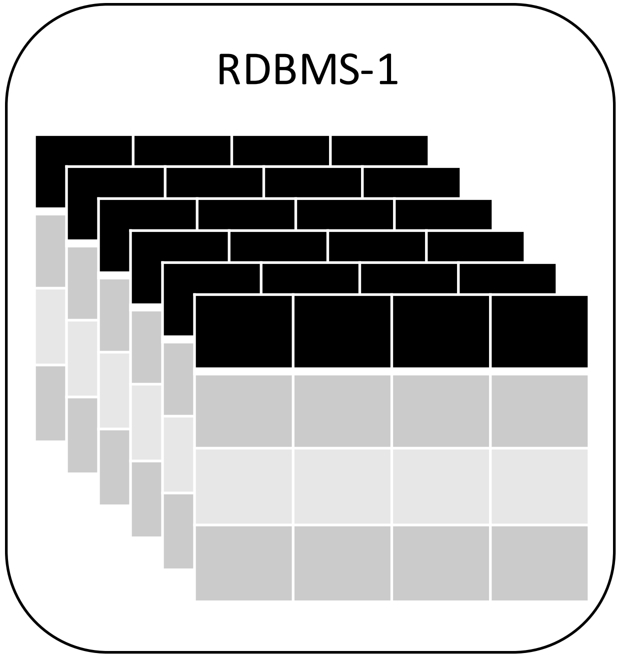
Department	Gender	Count
Finance	Male	1
Finance	Female	1
Tech	Female	1

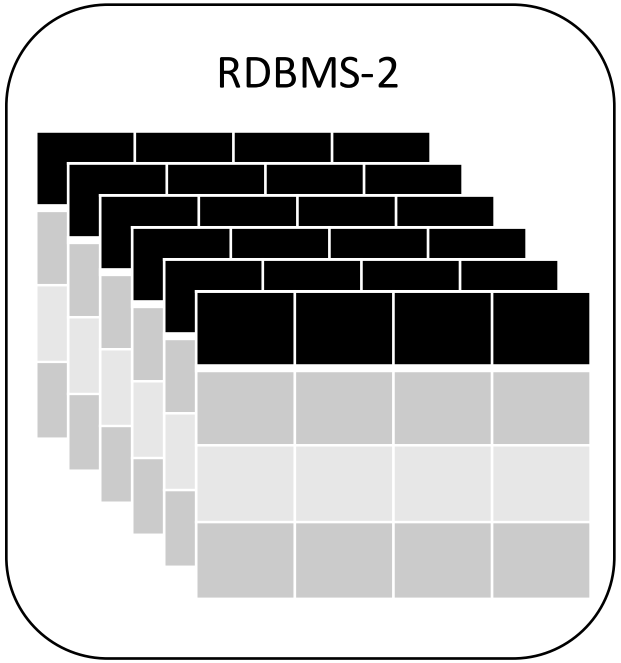
Employee Id	Name	Gender	Department
1001	John	Male	Finance
1002	Mary	Female	Tech
1003	Susan	Female	Finance

Employee gender distribution per department

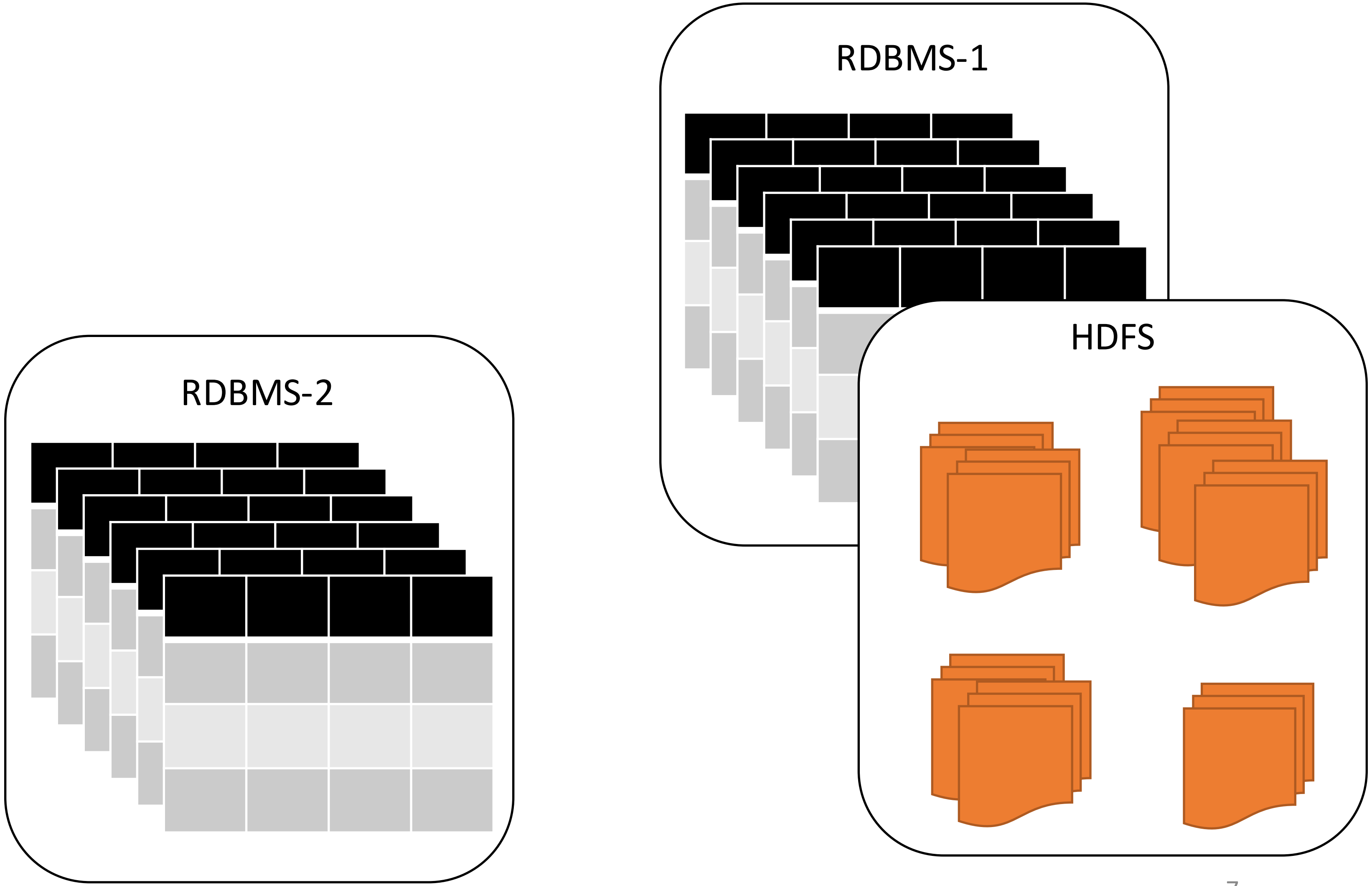


Employee gender distribution per department

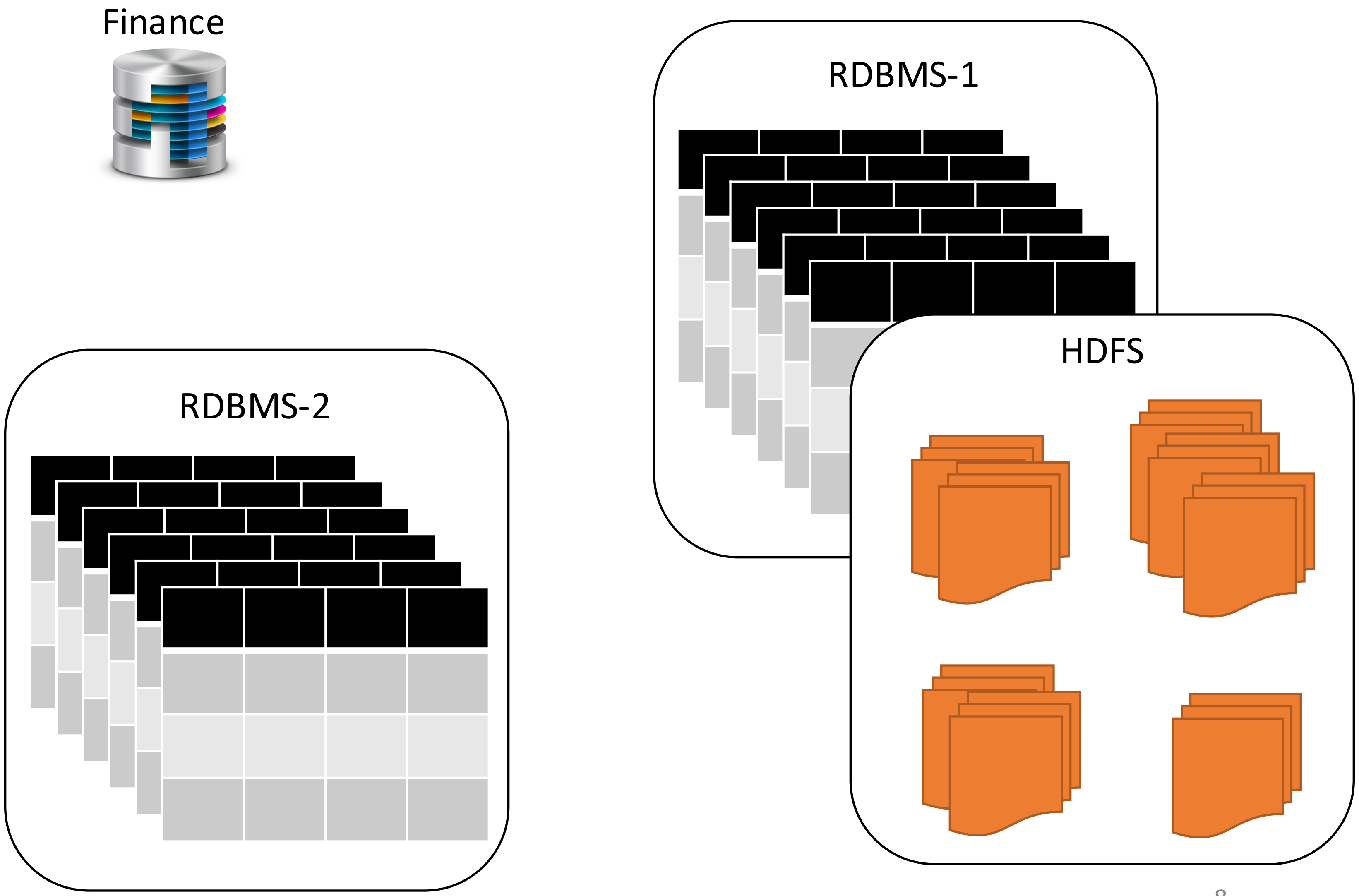




Employee gender distribution per department



Employee gender distribution per department



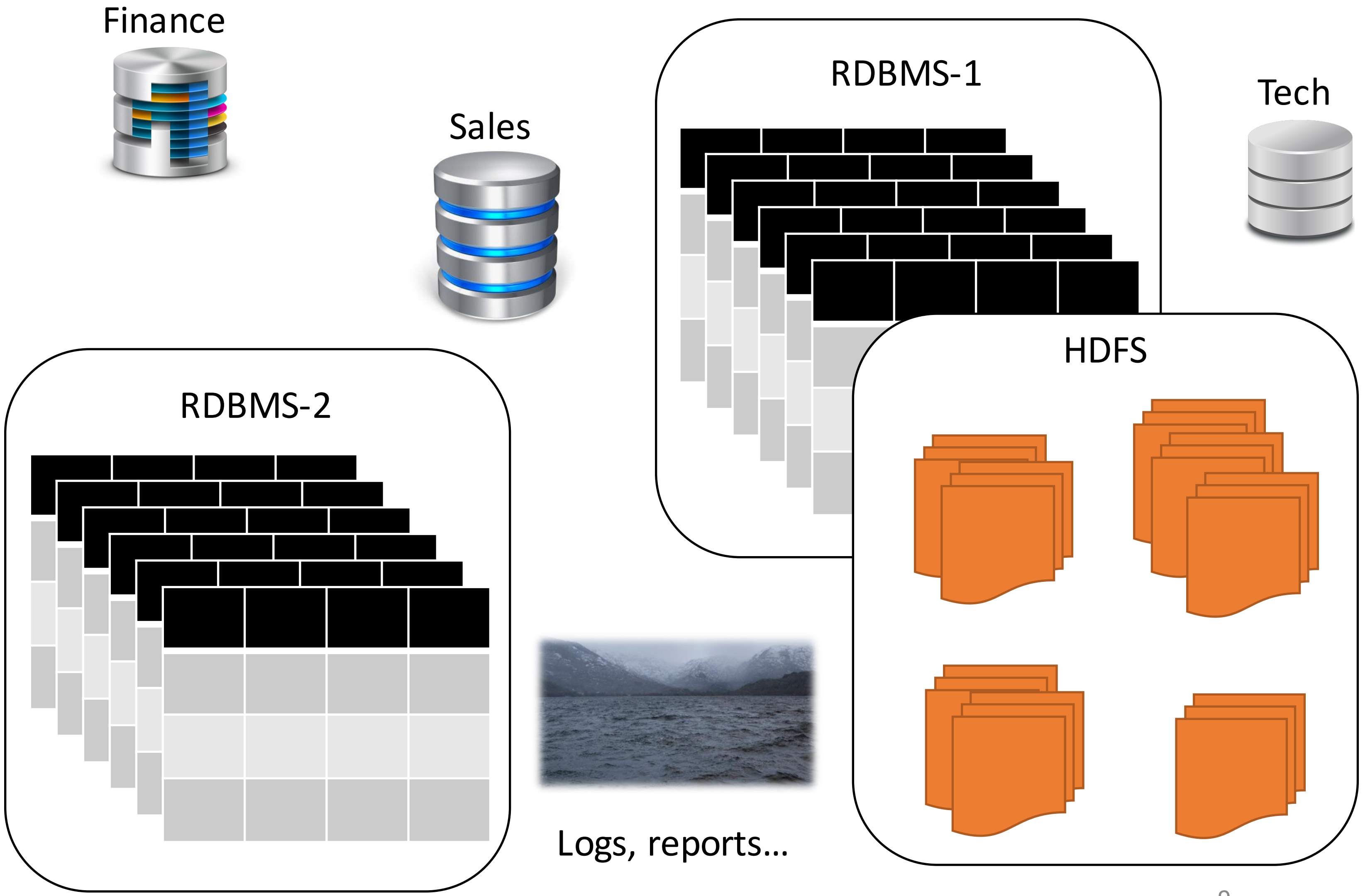
Finance

RDBMS-1

HDFS

RDBMS-2

Employee gender distribution per department



Finance

Sales

RDBMS-1

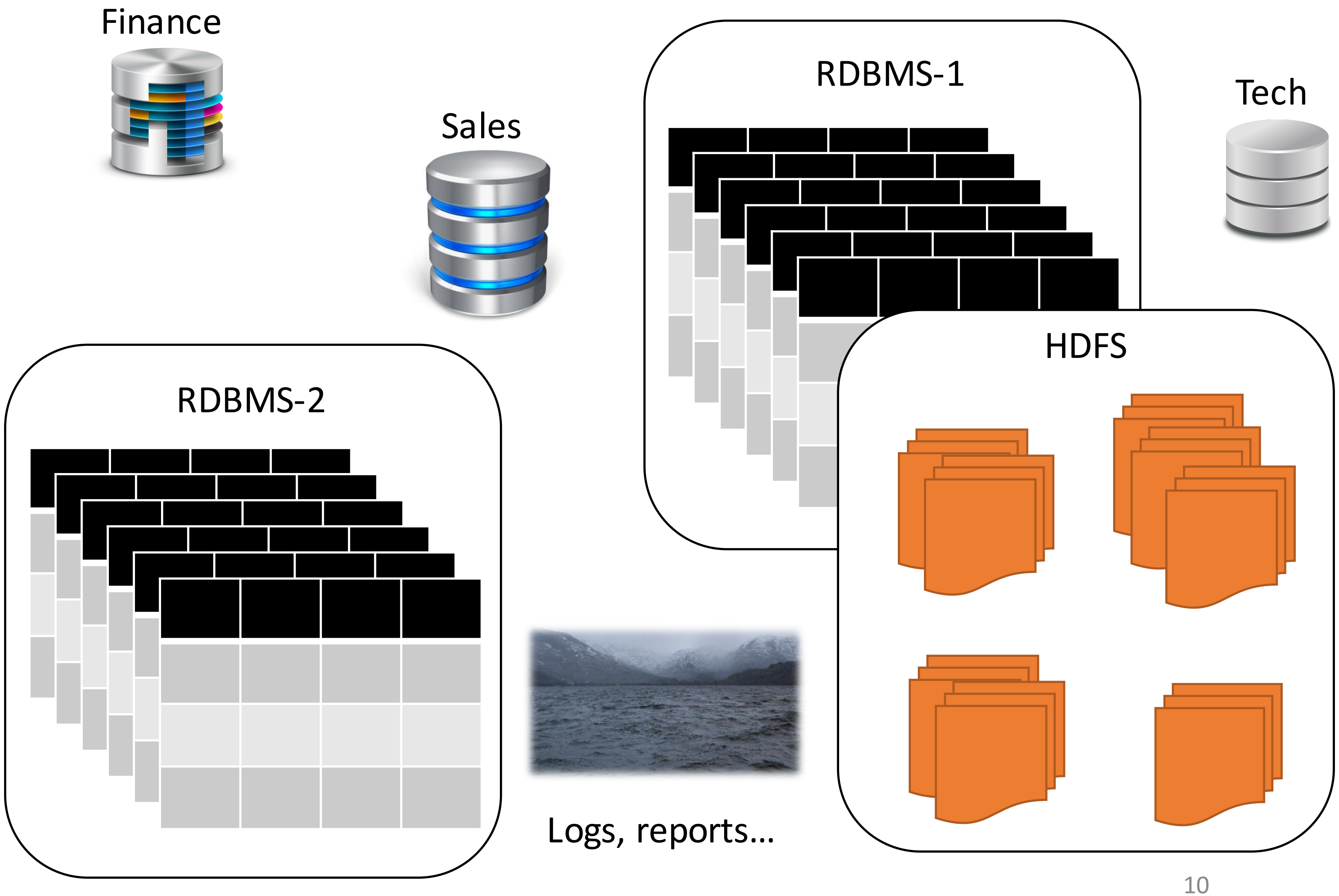
Tech

HDFS

RDBMS-2

Logs, reports…

Employee gender distribution per department



The Modern Company

- Data **heterogeneity**
  - Multiple departments and divisions
  - Data in relational tables, files in data lakes, reports in desktop machines…
- Increasing **data volumes**
  - Millions of tables, hundreds and thousands of databases

The Data Discovery Problem

- How do I **find relevant data** to the question at hand?

- Am I missing important data?

Finance

Sales

Tech

Logs, reports…

The De Facto Approach

- Social Networking
  - Ask other people:
    - " I heard Sam used to work with those datasets, ask him."
  - Not exhaustive
    - Sam:"Some of the datasets must be in the DBX database, the others I don't know…"

No single person in the organization knows about all datasets

What we really need: Declare what you want

```
SELECT department, gender, COUNT(*)
FROM Employee
GROUP BY gender, department;
```

Employee Id	Name	Gender	Department
1001	John	Male	Finance
1002	Mary	Female	Tech
1003	Susan	Female	Finance

What we really need: Declare what you want

```
SELECT department, gender, COUNT(*)
FROM Employee
GROUP BY gender, department;
```

Employee Id	Name	Gender	Department
1001	John	Male	Finance
1002	Mary	Female	Tech
1003	Susan	Female	Finance

$> **find_schema_with**(“*department*”, “*gender*”, “*employee*”)

What we really need: Flexibility

```
SELECT department, gender, year COUNT(*)
FROM Employee
GROUP BY gender, department, year;
```

Employee Id	Name	Gender	Department
1001	John	Male	Finance
1002	Mary	Female	Tech
1003	Susan	Female	Finance

What we really need: Flexibility

```
SELECT department, gender, year COUNT(*)
FROM Employee
GROUP BY gender, department, year;
```

Employee Id	Name	Gender	Department	Year
1001	John	Male	Finance	1987
1002	Mary	Female	Tech	1983
1003	Susan	Female	Finance	1988

$> **add_column(“*year*”)**

What we really need: Composable functions

I want to see instances of employees

```
$> search(“Raul Castro”)
```

What we really need: Composable functions

I want to see instances of employees

```
$> search("Raul Castro")
```

What fields have "*EmployeeID*"
or "*eid*" on their schema?

```
$> search_schema("EmployeeId")
   OR
   search_schema("eid")
```

What we really need: Composing functions

I want to see instances of employees

$> **search**("Raul Castro")

What fields have "*EmployeeID*"
or "*eid*" on their schema?

$> **search_schema**("EmployeeId")
OR
search_schema("eid")

Show me similar content to this table

$> **content_similar_to**(<table>)

Aurum: Data Discovery at Large

Aurum: Data Discovery at Large

- Motivation: The Data Discovery Problem
- **Aurum: Overview of the System**
- SRQL: Discovery Algebra
- Building the Metaschema Graph
    - Data profiling and summarization
    - Graph Builder component
- Roadmap

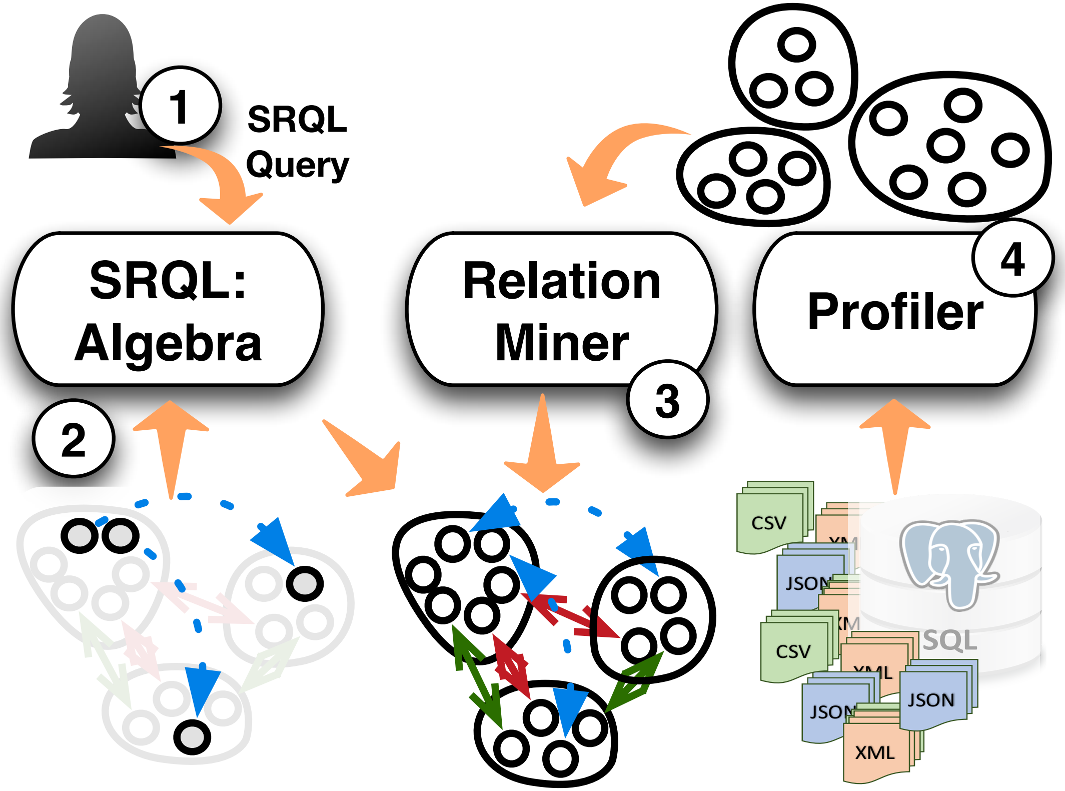
Aurum: Overview of the System

- **Observation**: *X is relevant to Y implies there exists a <u>relationship</u> between X and Y*

Aurum: Overview of the System

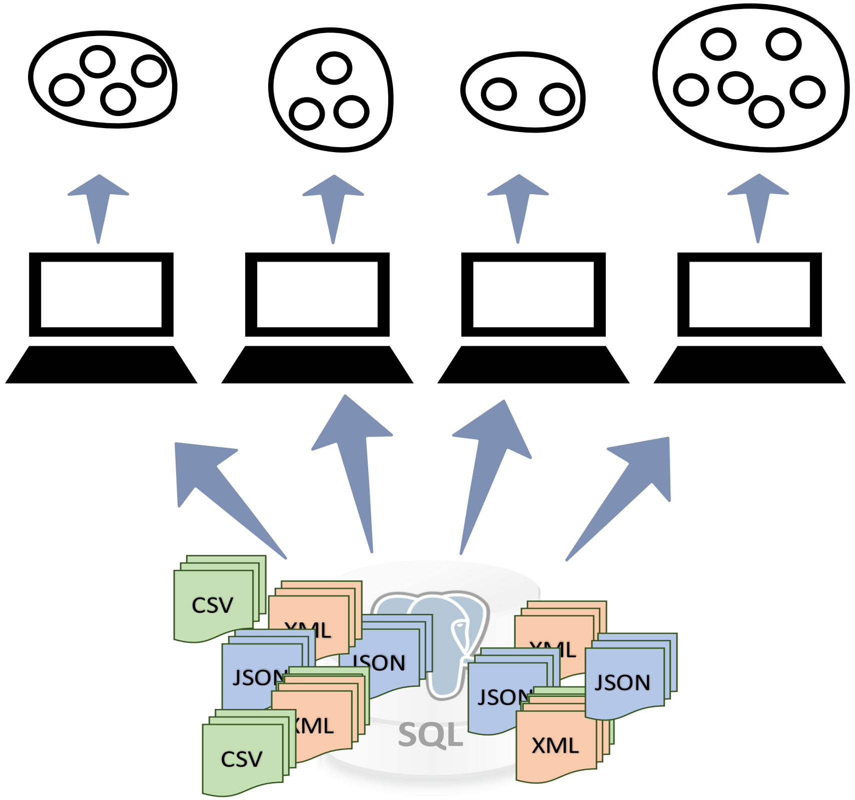
- **Observation**: *X is relevant to Y implies there exists a <u>relationship</u> between X and Y*

- SRQL algebra
  - Compose discovery queries
- Metaschema graph
  - Expose all relations in the data
- Profiler and Graph Builder
  - Summarize and build the graph fast



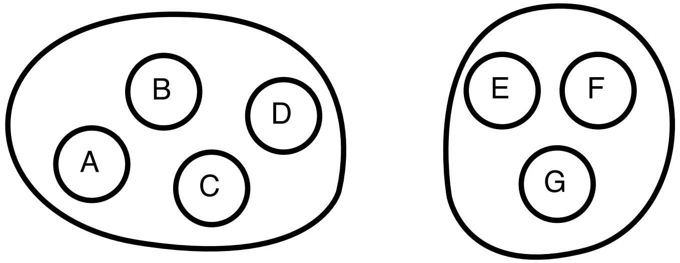
Profiler: Taming the Scale with Data Summaries

- **Goal:** Summarize *large* volumes of *heterogeneous* data
- High-Performance, scalable architecture
  - Read-once approach, sketches for O(n) operation
  - Distributed processing to scale to clusters
- A summary represents the **minimum discoverable element**



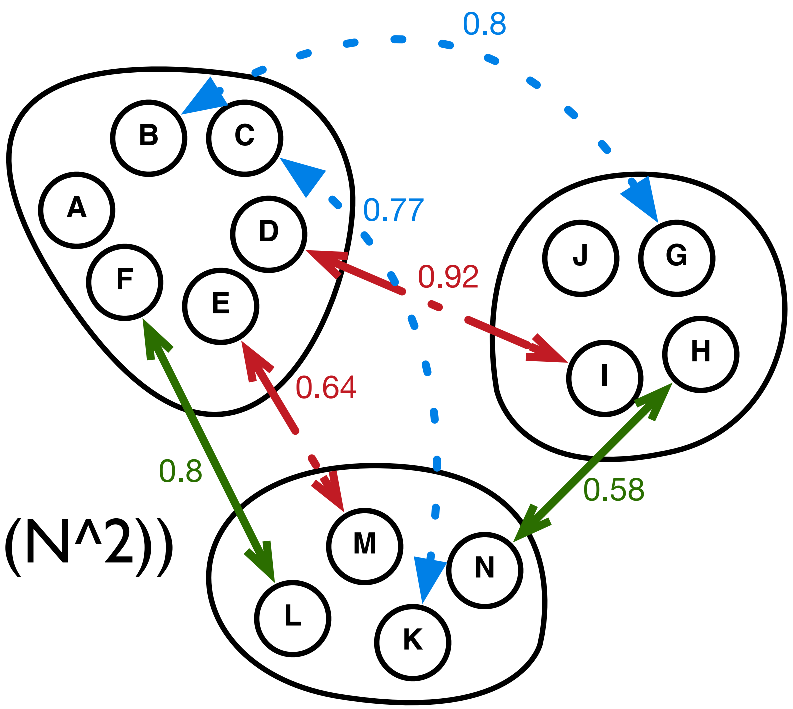
Profiler: Summary Grain and Content

- Minimum discoverable element: column
- Each summary is a **node** in the graph
  - Signatures: minhash, TF-IDF, or IQR+ - median
  - Data type, entity, cardinality, …
- Hierarchies represented with **hyperedges**
  - e.g., columns of a table



Graph Builder: Building the Metaschema Graph

- **Goal:** Mine relations of the underlying data
- **Edges** represent relations between nodes
  - Attribute name similarity, content similarity, PKFK…
- Scalable methods to extract relationships
  - LSH-hash signatures to avoid pairwise comparison (O(N^2))
  - e.g., minutes instead of weeks



SRQL: Discovery Algebra

- A SRQL query is a combination of ***data discovery primitives***
  - Lookup, edge, hyperedge, set and path primitives

similarTables(t: table) = schemaSim(t) AND contentSim(t)

SRQL: Discovery Algebra

- A SRQL query is a combination of ***data discovery primitives***
  - Lookup, edge, hyperedge, set and path primitives

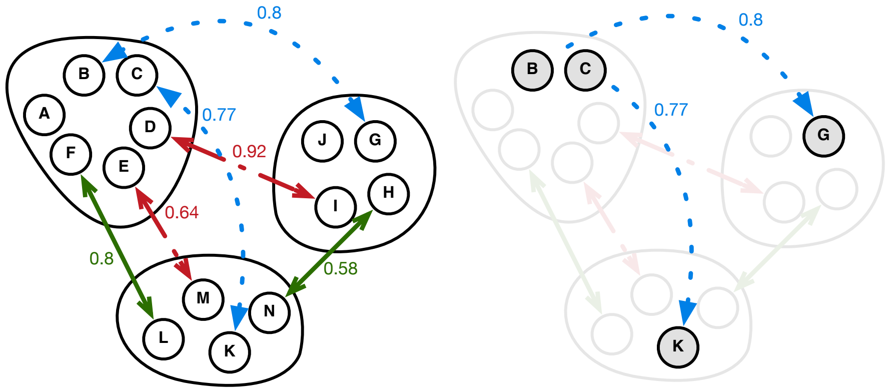
similarTables(t: table) = schemaSim(t) AND contentSim(t)

joinPath(src: table, tgt: table) = paths_between(src, tgt, Relation.PKFK)

SRQL: Discovery Algebra

- IR + Graph traversal to answer queries
  - **Goal:** Answer discovery queries in interactive times
- Ranking
  - Every user has a different **ranking criteria**
- Provenance
  - A SRQL query is a walk in the hypergraph
  - Explain results and debug queries!

Interactivity =
< 150ms



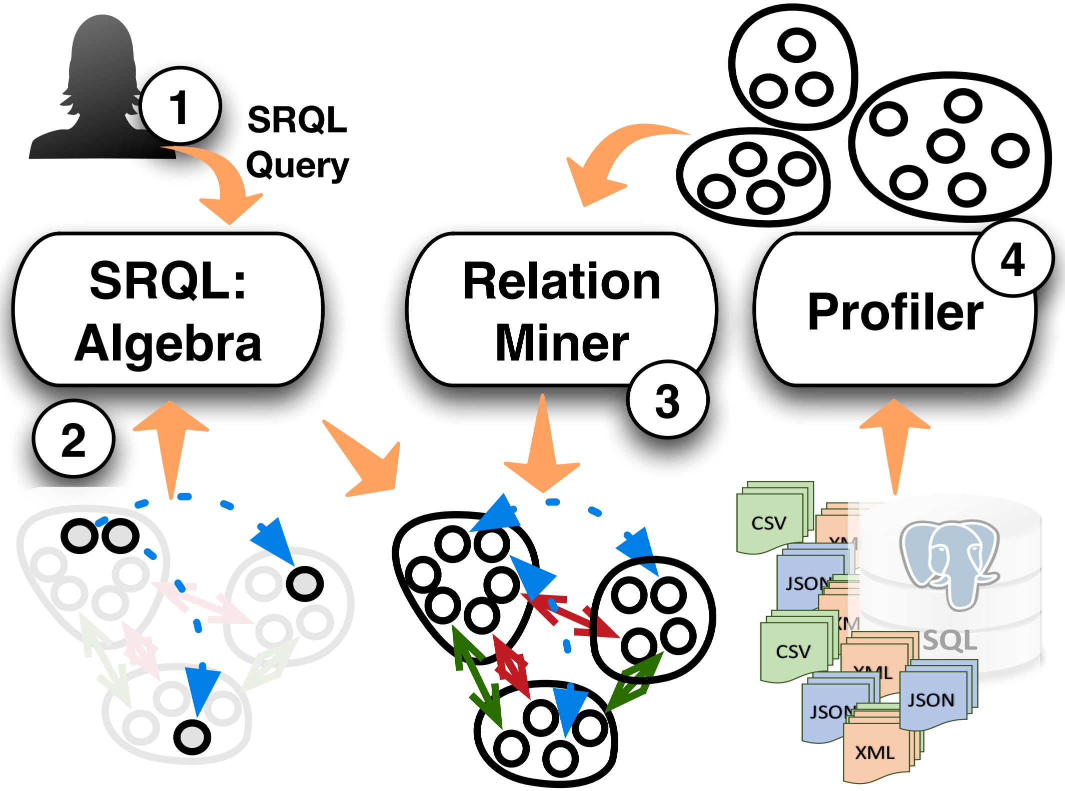
Outline

- Motivation: The Data Discovery Problem
- Aurum: Overview of the System
- SRQL: Discovery Algebra
- Building the Metaschema Graph
  - Data profiling and summarization
  - Graph Builder component
- **Roadmap**

Some Future Work

- Discovering **unstructured data**: PDFs, DOCs, HTML
  - How much unstructured data is in your organization?
- **Semantic** relations, inference
  - Is this behavior related to X?
- How well does my data solve my problems?
  - **Recommend** data, annotations and queries to users.

Aurum: A Large Scale Data Discovery System Based on Relation Retrieval Algebra



Raul Castro Fernandez

raulcf@csail.mit.edu

raulcastrofernandez.com