NoSQL

COSC 304 – Introduction to Database Systems

Relational Databases

Relational databases are the dominant form of database and apply to many data management problems.

Relational databases are not the only way to represent data and not the best way for some problems.

Other models:

- Hierarchical model
- Object-oriented
- JSON/XML
- Graphs
- Key-value stores
- Document models

Relational Databases Challenges

Some features of relational databases make them "challenging" for certain problems:

- 1) Fixed schemas – The schemas must be defined ahead of time, changes are difficult, and lots of real-world data is "messy".
  - Solution: Get rid of the schemas! Who wants to do that design work anyways? Will you miss them?
- 2) Complicated queries – SQL is declarative and powerful but may be overkill.
  - Solution: Simple query mechanisms and do a lot of work in code.
- 3) Transaction overhead – Not all data and query answers need to be perfect. "Close enough is sometimes good enough".
- 4) Scalability – Relational databases may not scale sufficiently to handle high data and query loads or this scalability comes with a very high cost.

NoSQL

NoSQL databases are useful for several problems not well-suited for relational databases with some typical features:

- Variable data: semi-structured, evolving, or has no schema
- Massive data: terabytes or petabytes of data from new applications (web analysis, sensors, social graphs)
- Parallelism: large data requires architectures to handle massive parallelism, scalability, and reliability
- Simpler queries: may not need full SQL expressiveness
- Relaxed consistency: more tolerant of errors, delays, or inconsistent results ("eventual consistency")
- **Easier/cheaper:** less initial cost to get started

NoSQL is not really about SQL but instead developing data management systems that are not relational.

- NoSQL – "Not Only SQL"

NoSQL Systems

There are a variety of systems that are not relational:

- MapReduce – useful for large scale analysis
- Key-value stores – ideal for retrieving specific data records from a large set of data
- Document stores – similar to key-value stores except value is a document in some form (e.g. JSON)
- Graph databases – represent data as graphs

MapReduce

MapReduce was invented by Google and has an open source implementation called Hadoop.

Data is stored in files. Users provide functions:

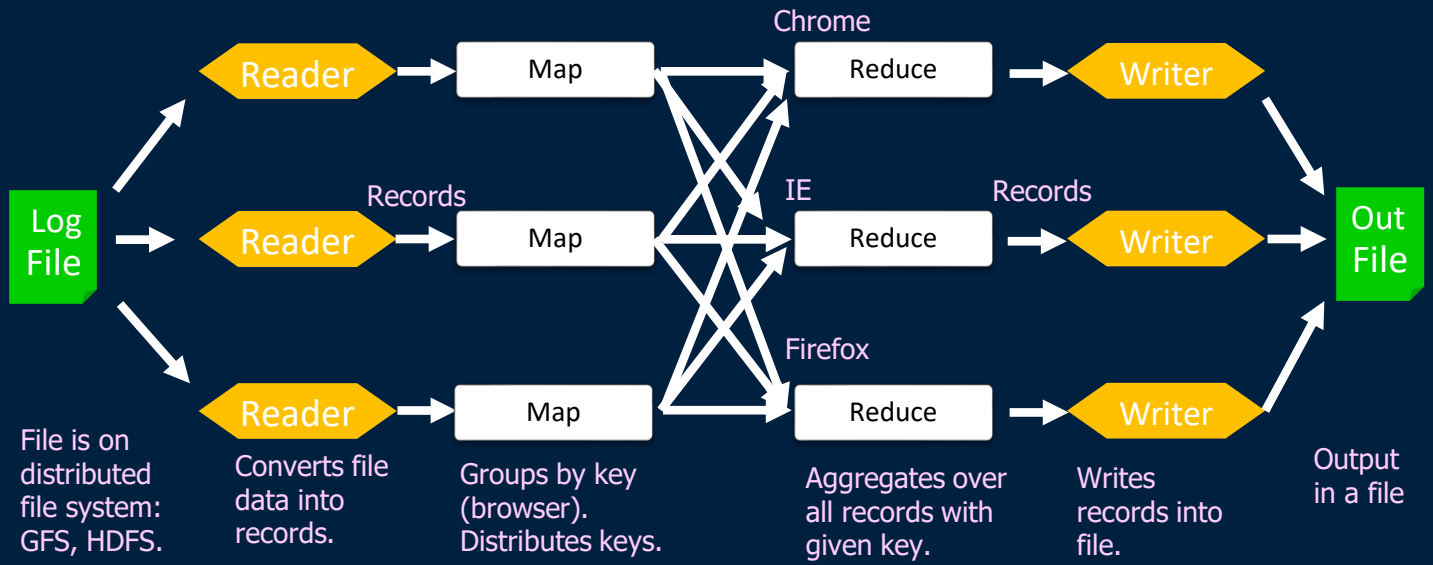
- reader(file) – converts file data into records
- map(records) – converts records into key-value pairs
- combine(key, list of values) – optional aggregation of pairs after map stage
- reduce(key, list of values) – summary on key values to produce output records
- write(file) – writes records to output file

MapReduce (Hadoop) provides infrastructure for tying everything together and distributing work across machines.

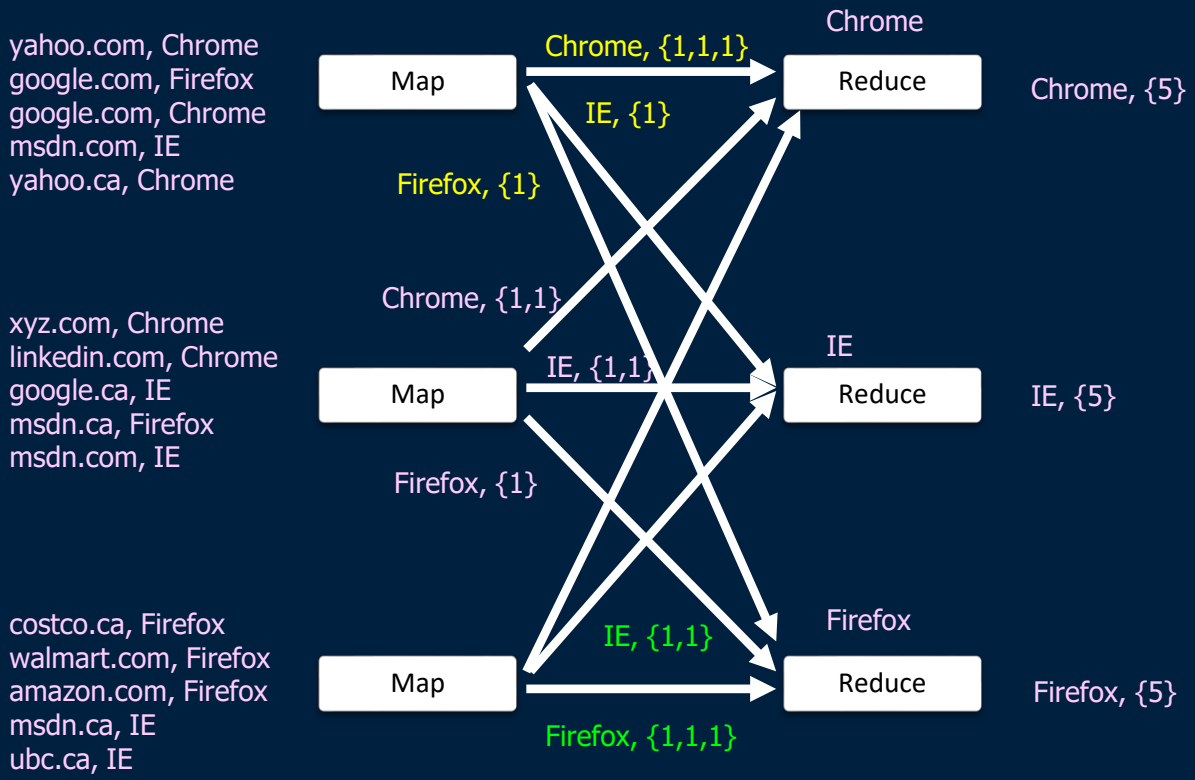
MapReduce Example
Web Data Analysis

Data file records: URL, timestamp, browser

Goal: Determine the most popular browser used.



MapReduce Example (2)
Web Data Analysis



MapReduce Extensions

The key benefit of MapReduce and Hadoop is their scalable performance, not that they do not support SQL. In fact, schemas and declarative SQL have many advantages.

Extensions to Hadoop combine the massive parallel processing with familiar SQL features:

- Hive – an SQL-like language variant
- Pig – similar to relational operators

Data manipulations expressed in these languages are then converted into a MapReduce workflow automatically.

Unlike relational databases, MapReduce systems handle failures during execution and will complete a query even if a server fails.

Key-Value Stores

Key-value stores store and retrieve data using keys. The data values are arbitrary. Designed for "web sized" data sets.

Operations:

- insert(key, value)
- fetch(key)
- update(key)
- delete(key)

Benefits: high-scalability, availability, and performance

Limitations: single record transactions, eventual consistency, simple query interface

Systems: Cassandra, Amazon Dynamo, Google BigTable, HBase

Document Stores

Document stores are similar to key-value stores but the value stored is a structured document (e.g. JSON, XML).

Can store and query documents by key as well as retrieve and filter documents by their properties.

Benefits: high-scalability, availability, and performance

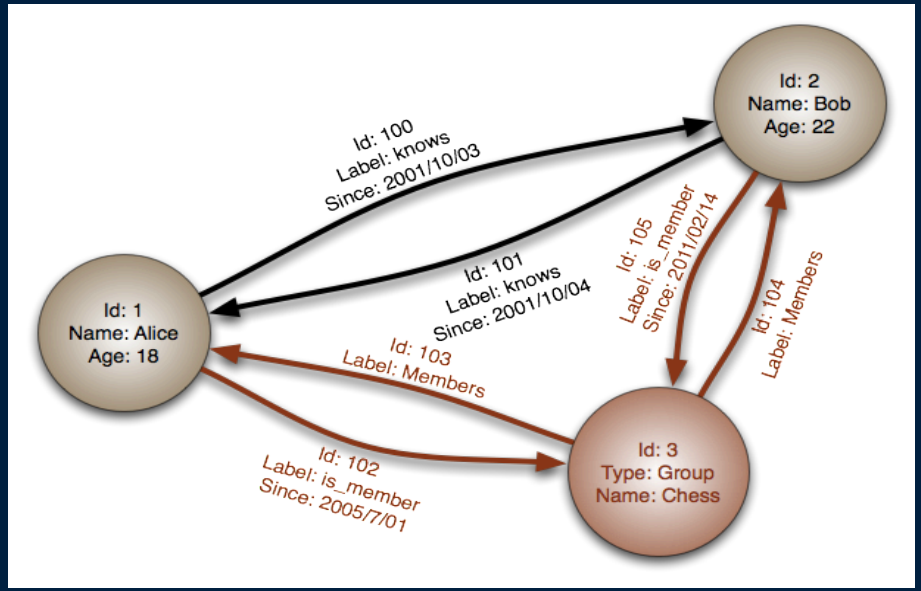
Limitations: same as key-value stores, may cause redundancy and more code to manipulate documents

Systems: MongoDB, CouchDB, SimpleDB

Graph Databases

Graph databases model data as nodes (with properties) and edges (with labels).

- Systems: Neo4J, FlockDB



NoSQL Question

Question: How many of the following statements are ***true***?

1) Neo4J is a document store.

2) Unlike Hadoop, a relational database is designed to restart any failed part of a query when a failure occurs.

3) Key-value stores are similar to a tree data structure.

4) SQL cannot be used to query MongoDB.

5) Relational systems typically scale better than NoSQL systems.

A) 0 **B)** 1 **C)** 2 **D)** 3 **E)** 4

Conclusion

NoSQL databases ("Not only SQL") is a category of data management systems that do not use the relational model.

There is a variety of NoSQL systems including:

- MapReduce systems
- Key-value stores
- Document stores
- Graph databases

NoSQL databases are designed for high performance, availability, and scalability at the compromise of restricted query languages and weaker consistency guarantees.

Objectives

- Understand alternative models for representing data besides the relational model
- List some NoSQL databases and reason about their benefits and issues compared to the relational model for certain problems
- Explain at a high-level how MapReduce works


UBC
THE UNIVERSITY OF BRITISH COLUMBIA