Distributed Big Data Frameworks
A Panorama

LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 809965.

BIG DATA & AI LANDSCAPE 2018

Final 2018 version, updated 07/15/2018

mattturck.com/bigdata2018

FIRSTMARK
EARLY STAGE VENTURE CAPITAL

THANK YOU !

LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 809965.

Database Paradigms

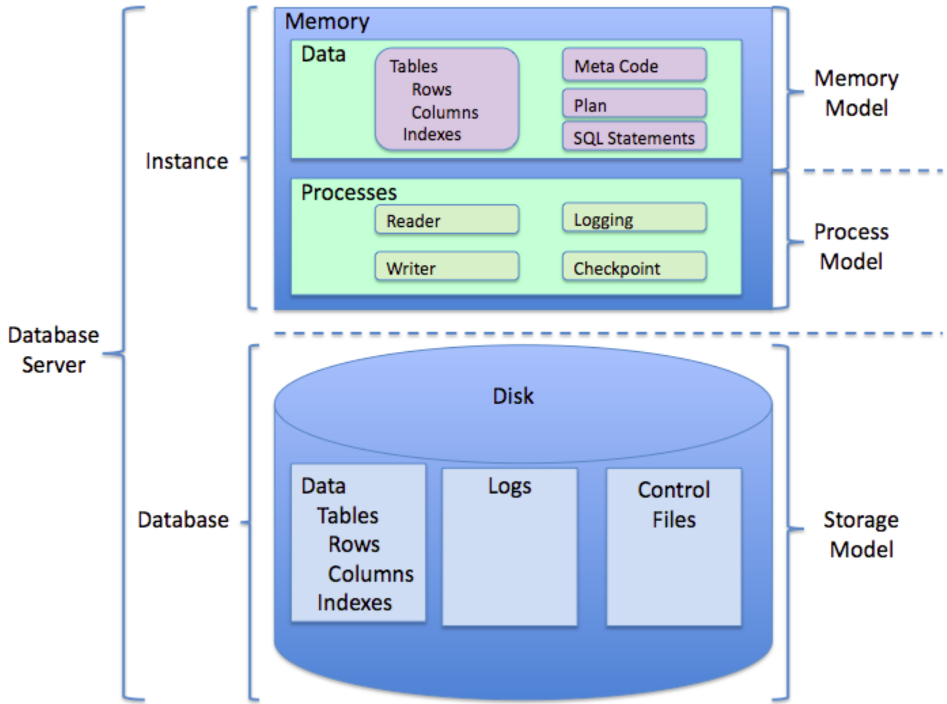
Database Paradigms

- **Relational (RDBMS) — The SQL world…!**

Relational (a quick reminder)

- ACID set of properties *i.e*. atomicity, consistency, isolation and durability.
- SQL is the canonical query language
- MySQL, PostgreSQL, Oracle, …

Relational (a quick reminder)



https://en.wikipedia.org/wiki/Relational_database#/media/File:RDBMS_structure.png

Database Paradigms

- Relational (RDBMS) — The SQL world…!

- NoSQL
  - **Key-Value stores**

Key-Value stores

- Paradigm → One key = One value
  - Without duplicate
  - Usually sorted
- Key is like a hash
- Value is seen as a binary object

- Examples:
  - Amazon Dynamo
  - MemcacheDB
  - Apache Accumulo

Key-Value stores

Key	Value (Opaque)
User:2:friends	{23, 76, 233, 11}
User:2:settings	Theme: dark, cookies: false
User:3:friends	[234, 3466, 86, 55]

Database Paradigms

- Relational (RDBMS) — The SQL world…!
- NoSQL
  - Key-Value stores
  - **Document databases**

Document databases

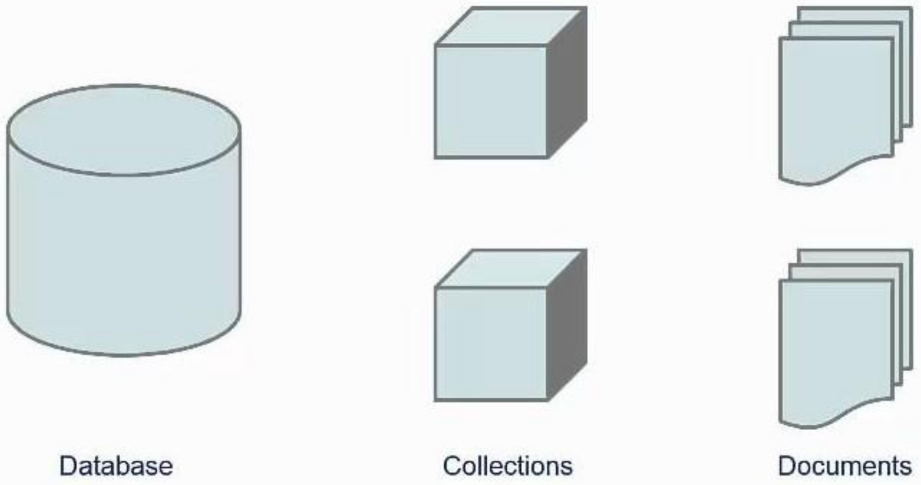
- Key-Value store, but the value is structured and *understood* by the DB.

- Querying data is possible (by other means than just a key).

- Examples:
  - Amazon SimpleDB
  - CouchDB
  - Riak
  - **MongoDB**

Document databases

Documents are gathered together in collections within the database.



Database Paradigms

- Relational (RDBMS) — The SQL world…!

- NoSQL
  - Key-Value stores
  - Document databases
  - **Wide Column stores (*e.g.* BigTable and its variations)**

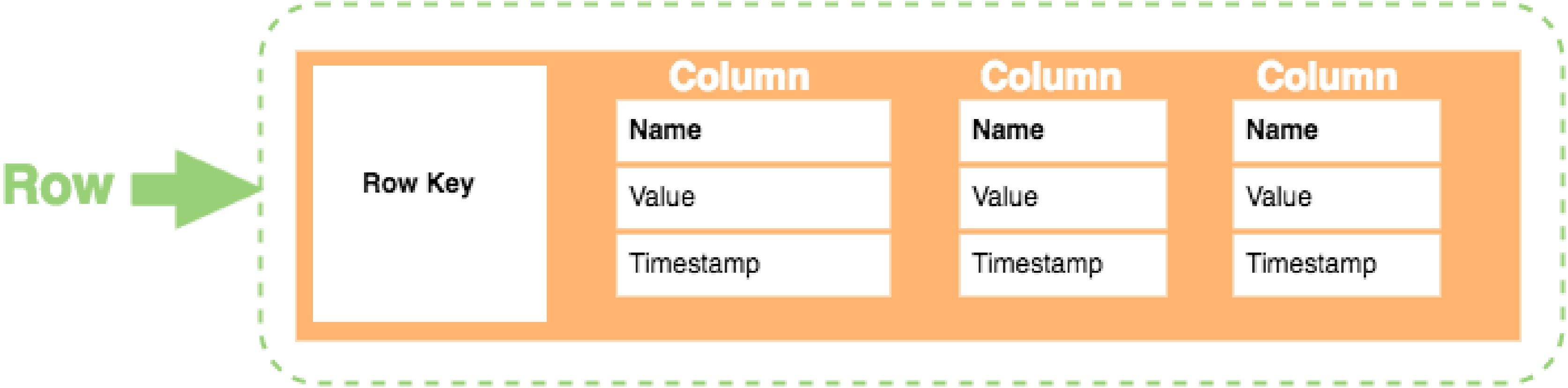
Wide Column stores

- Often referred as BigTables systems

- “Sparse, distributed mutli-dimensional sorted map”

- Examples:
  – Google BigTable
  – Cassandra (Facebook)
  – HBase

Wide Column stores



https://database.guide/wp-content/uploads/2016/06/wide_column_store_database_example_row-1.png

Database Paradigms

- Relational (RDBMS) — The SQL world…!

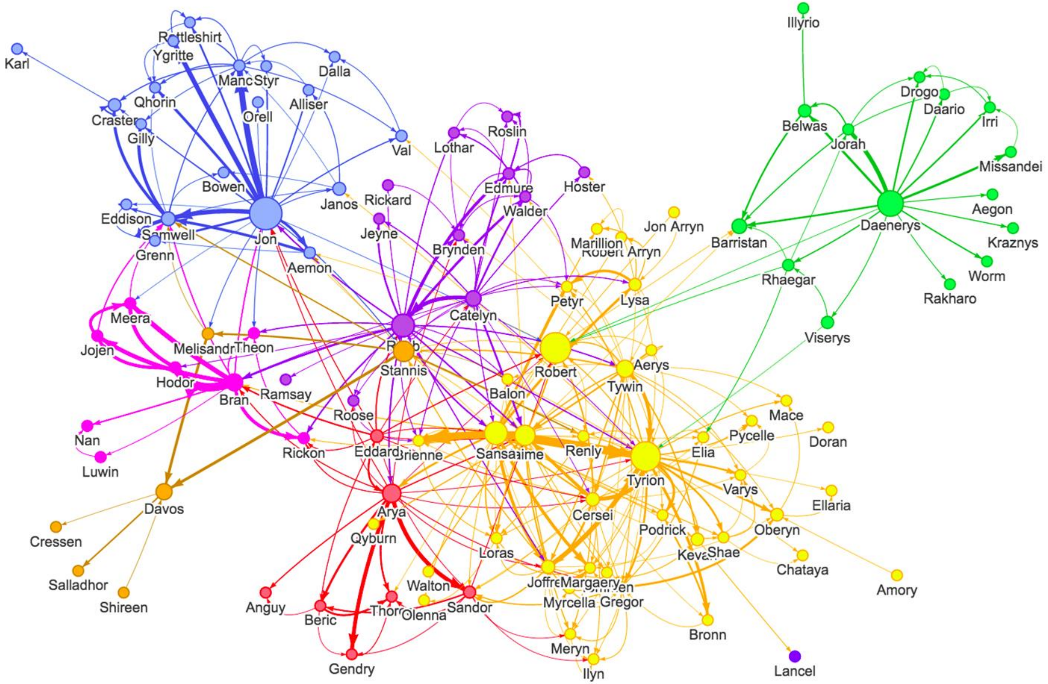
- NoSQL
  - Key-Value stores
  - Document databases
  - Wide Column stores (*e.g.* BigTable and its variations)
  - **Graph databases**

- Other ones…

Graph databases

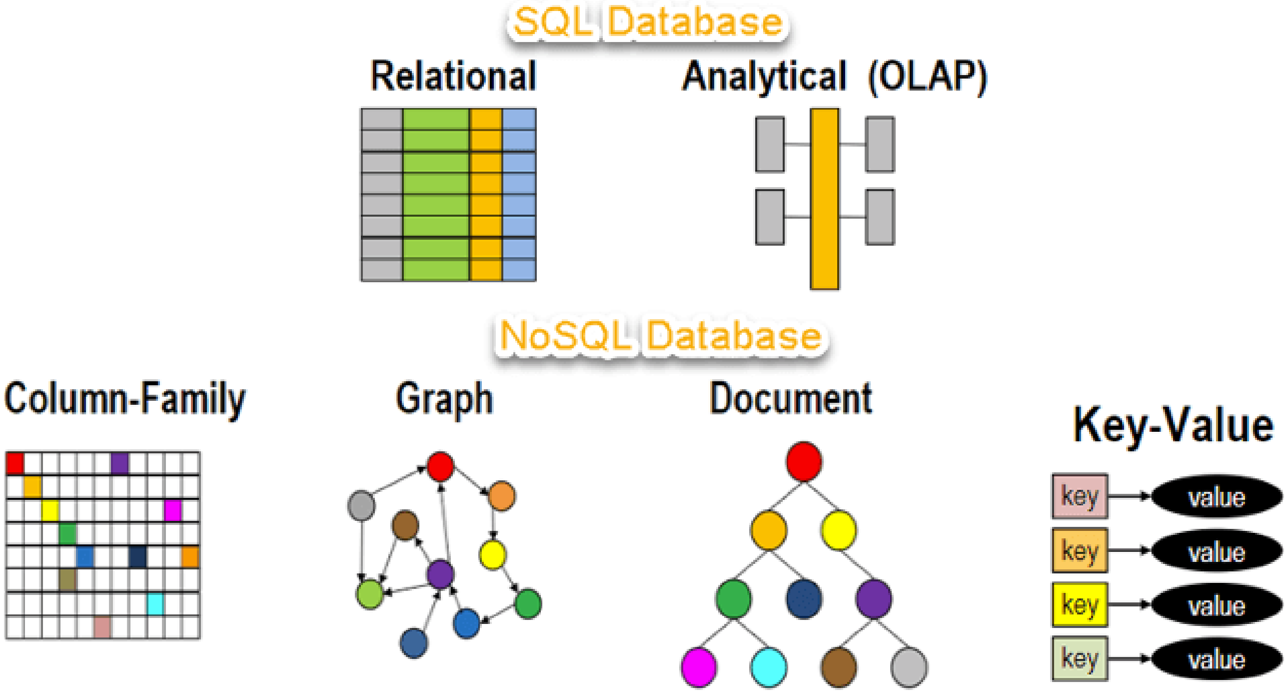
- Multi-relational graph
- Put emphasis on links between data pieces
- Examples:
  - Neo4j
  - InfoGrid
  - Triplestores…

Graph databases



https://neo4j.com/blog/graph-of-thrones/

Database Paradigms (Visually)



https://www.guru99.com/nosql-tutorial.html

Selected Storage Systems

MongoDB

- Document database (NoSQL)
  - scalability and flexibility
  - querying and indexing
- Stores data in
  - JSON-like documents
  - schema free database
- Open-Source

What is MongoDB great for?

- RDBMS replacement for Web Applications
- Semi-structured Content Management
- Real-time Analytics & High-Speed Logging
- Caching and Scalability

Apache Hive

- Apache Hive is a data warehouse
  - Developed by Facebook
  - On top of Apache Hadoop
- Provides
  - Data summarization
  - Query
  - Analysis
- Open-Source
- Gives an SQL-like interface

Apache Hive - Some Facts

Most Queries Per Hour

100,000 Queries Per Hour
(Yahoo Japan)

Analytics Performance

100 Million rows/s Per Node
(with Hive LLAP)

Largest Hive Warehouse

300+ PB Raw Storage
(Facebook)

Largest Cluster

4,500+ Nodes
(Yahoo)

https://www.slideshare.net/MahmoodRezaEsmailiZa/apache-hive-introduction

Apache Hive - Limitations

- **Not** design for online transaction processing
- **Not** suited for real-time queries
- **Not** made for low-latency query
- Certain standard SQL functions do not exist
  - `NOT IN`
  - `NOT LIKE`
  - `NOT EQUAL`

Apache Cassandra

- Facebook inbox search feature
- millisecond read and write times
- Designed for linear, incremental scalability on top of commodity hardware.
- Open-Source since 2008

Cassandra - Strenghts

- Linear scale performance when adding node
- Peer-to-peer architecture instead of master-slave
- Fault tolerant in case of node failure
- High performance
- Schema-free/Schema-less

Cassandra - Limitations

Use cases where it is better to avoid using Cassandra

- If there are too many joins required
- To store configuration data
- During compaction, things slow down
- Aggregation Operator support is limited
- Can update or delete data but not designed for

Distributed Stream Processing

Apache Kafka

- Distributed event streaming platform
- Able to handle up to trillions of events a day
- Initially conceived as a messaging queue
- Open-sourced by LinkedIn in 2011
- Useful for:
  - Stream processing
  - Website activity tracking
  - Metrics collection and monitoring
  - Log aggregation

Apache Kafka

Three key capabilities

- Publish and Subscribe
  - Stream of records
- Process
  - Streams of records as they occur
- Store
  - Streams of records in fault tolerant manner

Search, Indexing, visualization

ElasticSearch

- Distributed and highly available search engine
  - Indexes are sharded (replicas)
  - read/search on any replica shards
- Multi-tenant
  - Support for more than one index
- Various set of APIs
- Document oriented
- Reliable
- Near real time search

ElasticSearch

elasticsearch

- Data into ElasticSearch
  - Receive data in form of JSON documents
  - Ingest data using Logstash
  - Connectors to other data stores
- Stores and add searchable reference to the document
- All data is indexed by default
- Every field has a dedicated inverted index
- All of inverted indexes can be used in a query

Visualization

Kibana

- Browser-based analytics and search dashboard for Elasticsearch
  - search
  - view
  - interact

  with the data stored in Elasticsearch indices
- Visualize data
  - Charts
  - Tables
  - Maps

Kibana

- Display changes in real time
- Discover
  - Explore data using selected index patterns
- Visualize
  - Create visualizations of data based on
    - ElasticSearch queries, Search saved from Discover
  - Stored as dashboards
- Dashboard
  - collection of visualizations and searches

Processing Frameworks

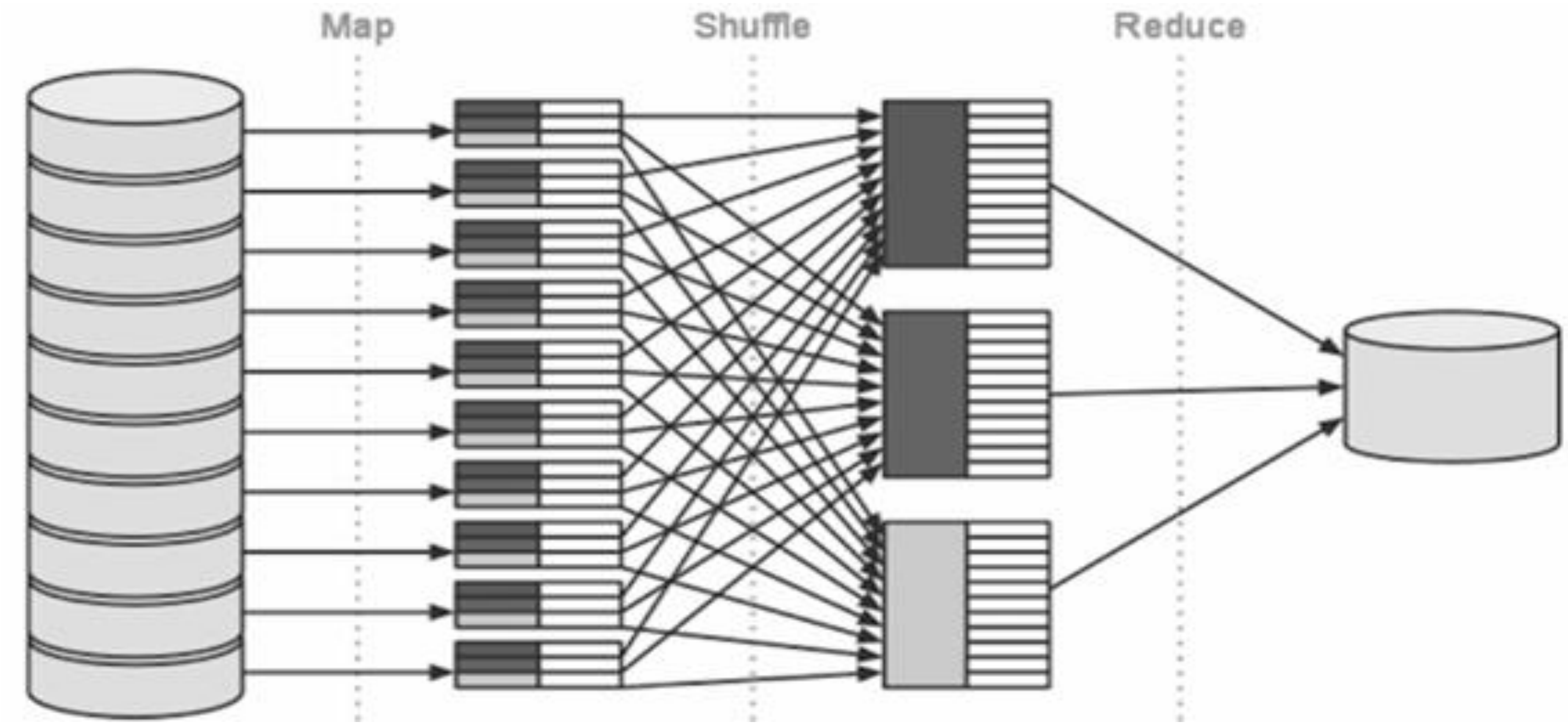
Analytic Frameworks

- Batch-only
  - Apache Hadoop MapReduce
- Stream and In-Memory Computing
  - Apache Spark
  - Apache Flink

Hadoop MapReduce

- Distributed framework to process vast amounts of data (multi-terabyte data-sets)
  - Cluster of commodity hardware
  - Reliable
  - Fault tolerant
- MapReduce job
  - Divides the large data into independent chunks
  - Processed by Map-tasks in parallel
  - Sorted output is passed to the reduce-tasks
  - Typically both input and output are stored in filesystem (HDFS)

MapReduce Engine



https://www.dummies.com/wp-content/uploads/473708.image0.jpg

Map reduce

- First popular data flow model
- In the Map-Reduce model, the user provides two functions (map and reduce)
  - Map() must output key-value pairs
  - Input to the reduce is partitioned by key across machines (shuffled)
  - reduce() output the aggregated values

Processing of Map tasks

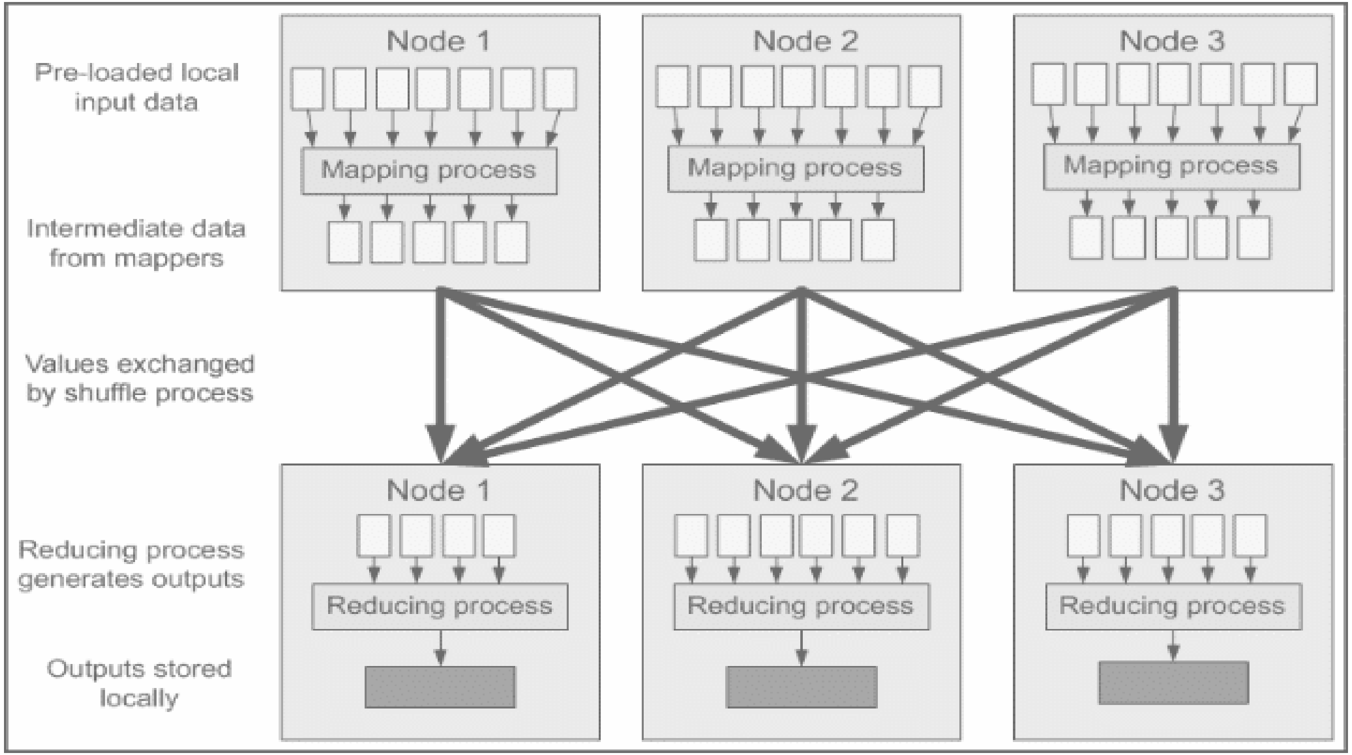
- Given a file divided into multiple parts (splits).
- Each record (line) is processed by a Map function,
  - written by the user,
  - takes an input key/value pair
  - produces a set of intermediate key/value pairs.
  - e.g. (doc—id, doc-content)
- Draws an analogy to SQL group-by clause

Processing of Reduce Tasks

Given a set of (key, value) records by map tasks

- The intermediate values are combined into a list based on keys and given to a reducer.
- Each reducer further performs an *aggregate* function (e.g., average) computed over all the rows with the same "key"

Visualizing map and reduce tasks



https://cdn.intellipaat.com/mediaFiles/2015/07/hadoop-mapreduce1.jpg.png

Example: Word counting in class

”Consider the problem of counting the number of occurrences of each word in a large collection of documents”

Distribute — Map — Shuffle — Reduce

Distribute: Divide a collection of documents among the class

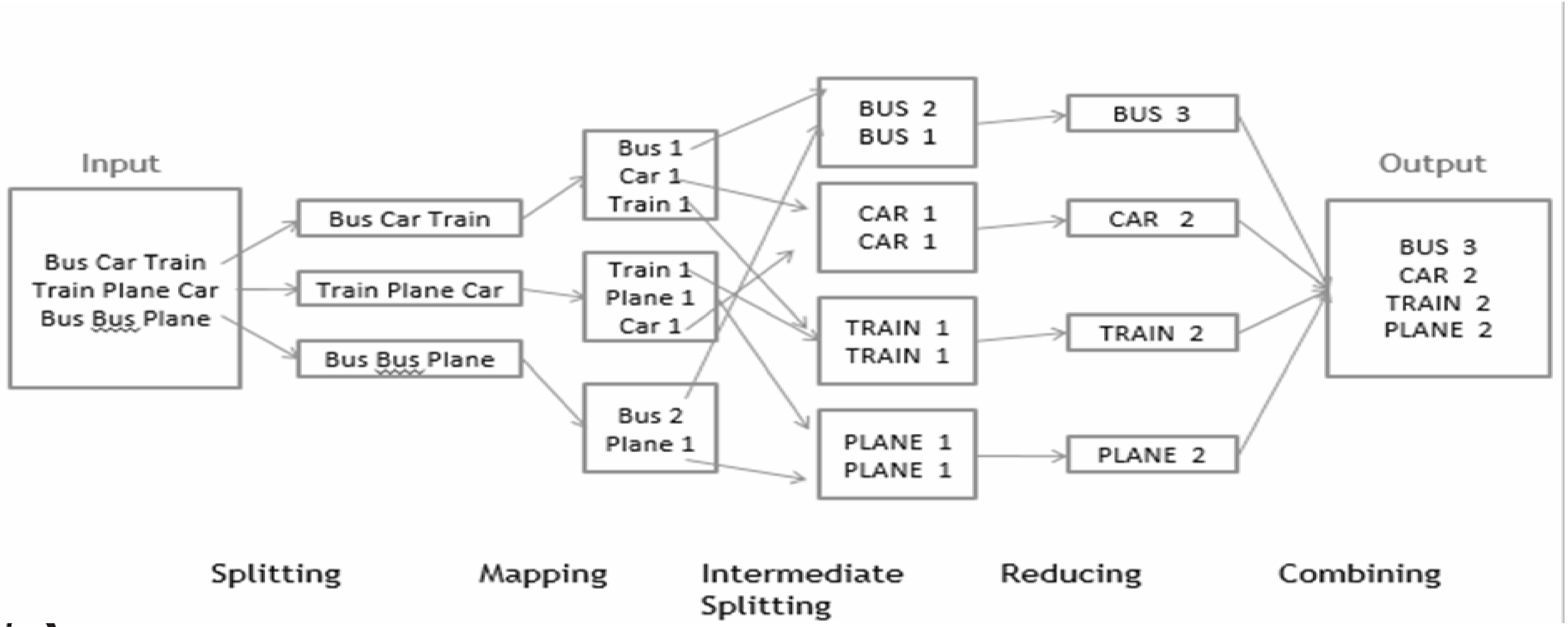
Map: Each person calculates counts of individual words in the documents

independent

Shuffle: Gather the words and counts

Reduce: Sum up the counts from all the documents for all the words

Word Count MapReduce



https://dzone.com/storage/temp/1329325-111.png

Drawbacks of MapReduce

- Forces data analysis workflow into a map and a reduce phase
  - You might need
    - Join
    - Filter
    - Sample
  - Complex workflows that do not fit into map/Reduce
  - Mapreduce relies on reading data from disk
    - Performance bottleneck
    - Especially for iterative algorithms
    - e.g. Machine Learning

Requirement . .

- A tool, compatible with the existing environment
- Without replacing the stack
  - replace map-reduce only
- Generic
  - Provides a rich API, more functions
- Reduces Disk I/O
  - Faster
  - In-memory computations
- Provides an interactive shell

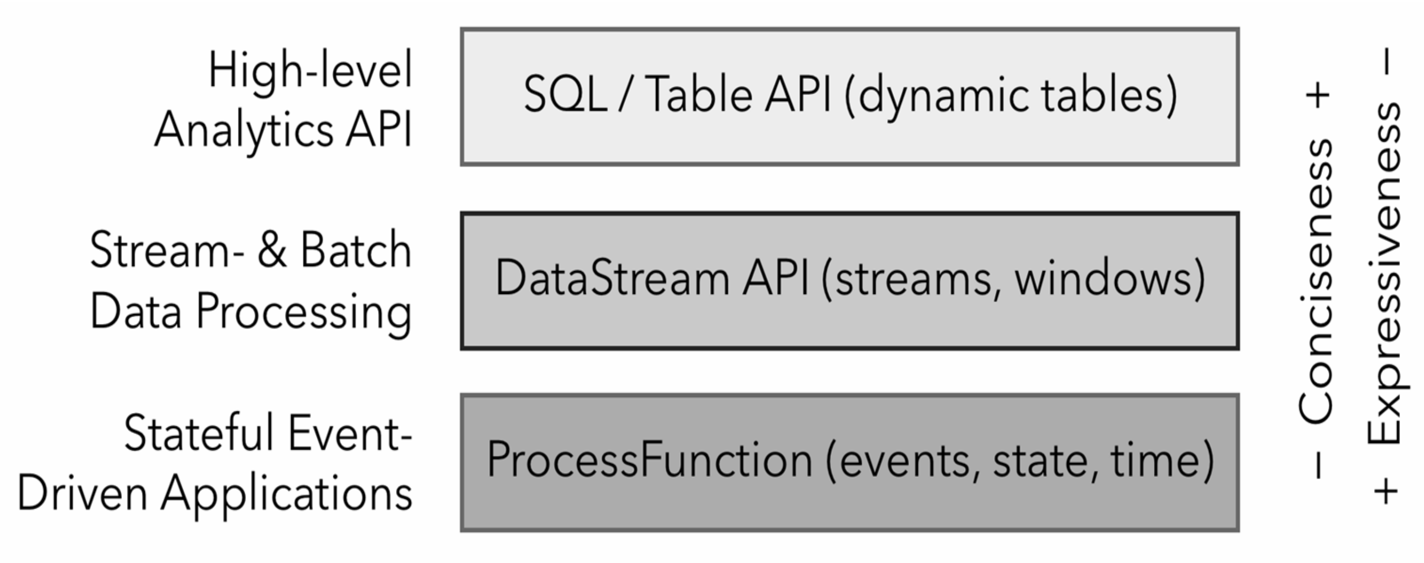
Apache Flink

- Distributed stream processing engine
- Process bounded and unbounded streams
- Generic deployment
- Scalable
  - Trillions of events
  - Terabytes of states
  - Thousands of cores
- In-Memory performance

Apache Flink APIs

- Data Stream API
  - bounded or unbounded streams of data
- Dataset API
  - bounded data sets
  - Transformations (Filter, map, join)
- Table API
  - SQL-like expression language for relational stream

Apache Flink Layered APIs



Apache Flink Libraries

- Complex Event Processing (CEP)
  - Pattern detection from events
- DataSet API
  - Map, Reduce, join, iterate
- Gelly
  - Scalable Graph processing library

References

https://flink.apache.org/

Dr. Damien Graux
damien.graux@iais.fraunhofer.de

Dr. Hajira Jabeen
jabeen@cs.uni-bonn.de

THANK YOU !

LEARNING, APPLYING, MULTIPLYING BIG DATA ANALYTICS

This project has received funding from the European Union's Horizon 2020 Research and Innovation programme under grant agreement No 809965.