FIT1043 Introduction to Data Science

Module 3:

Data Types and Storage

Lecture 6

Monash University

Unit Schedule: Modules

Module	Week	Content
1.	1	overview and look at projects
	2	(job) roles, and the impact
2.	3	data business models
	4	application areas and case studies
3.	5	characterising data and "big" data
	6	**processing big data ~~sources and case studies~~**
4.	7	resources and standards
	8	resources case studies
5.	9	data analysis theory
	10	data analysis process
6.	11	issues in data management
	12	data management frameworks

Discussion: R

- Powerful language for visualising and building predictive models of data
- Very easy to use with lots of inbuilt functionality.
- Great for exploratory data analysis
- Not as scalable as programming languages: Java, Python,

Discussion: Assessment

- First assignment due in Week 8.
- Are there any questions about the assignment?

Big Data Processing
(ePub section 3.4)

processing data at scale, especially for analysis

- databases
  - storing and accessing data
- distributed processing
  - breaking up computation to scale it up

Business Context

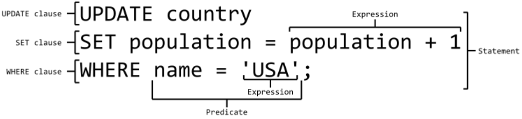
- businesses function in a continuously changing environment:
  - fixed formats as per Relational Database Management System (RDBMS) not suitable
- businesses function in a continuously changing environment:
  - usage varies, requires complex analytical queries
- need to reach insights faster and act on them in real time
  - stream processing

Big Data Processing: Databases

storing and accessing data

SQL Review

- Relational Database Management Systems (RDBMS)
- SQL ::= structured query language



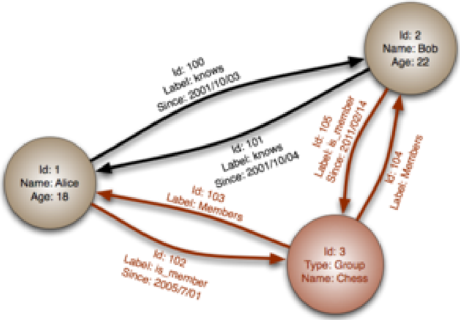
- rather like large scale set of Excel spreadsheets with better indexing and retrieval
- transaction oriented with support for throughput, correctness, distribution, ...

JSON Example


```json
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```


- example from Wikipedia
- no fixed format
- semi-structured, key-value pairs, hierarchical
- "friendly" alternative to XML
- self-documenting structure
- example, *EventRegistry file*

Graph Database Example



- example graph
- example content
  FreeBase page for "Arnold Schwarzenegger"
- example content format *FreeBase extract*
- stores graph, commonly as triples, `subject, verb, object`
- commonly used to store Linked Open Data

Database Background Concepts

in-database analytics: the analytics is done within the DB

in-memory database: the DB content resides memory

cache: data stored in-memory

key-value: *value* accessible by *key*, *e.g.*, hash table

information silo: an insular information system incapable of reciprocal operation with other, related information systems

- if two big banks merge, then initially their RDBMSs will be siloed
- in a big insurance company, auto and home insurance customer RDBMSs may be siloed

Database Background Concepts, cont.

Many NoSQL and SQL DBs offer:

- large scale, distributed processing
- robustness
- general query languages
- some notion of consistency

  e.g. "eventually" as nodes spread updates

Beyond SQL Databases

Type	Examples	Notes
RDBMS	*MySQL*, *MSSQL Server*	SQL
Object DB	*Zope*, *Objectivity*	navigate network
Doc. DB	*MongoDB*, *CouchDB*	JSON like, Javascript like queries
key-val cache	*Memcached*, *Coherence*	in-memory
key-val store	*Aerospike*, *HyperDex*	not in-memory but highly optimised
tabular key-val	*Cassandra*, *HBase*	relational-like, "wide column store"
graph DB	*Neo4j*, *OrientDB*	RDF, SPARQL,

Beyond SQL Databases (NoSQL)

- NoSQL databases offer a rich variety beyond traditional relational.
- Many target web applications.
- See blog post by Eric Knorr 19/11/2012 on `Infoworld.com`, *"The wild, crazy world of databases"*
- See blog post by Fabian Pascal 12/17/2015 on `AllAnalytics.com`, *"Data Fundamentals for Analysts: Documents and Databases"*.

Overview: Databases

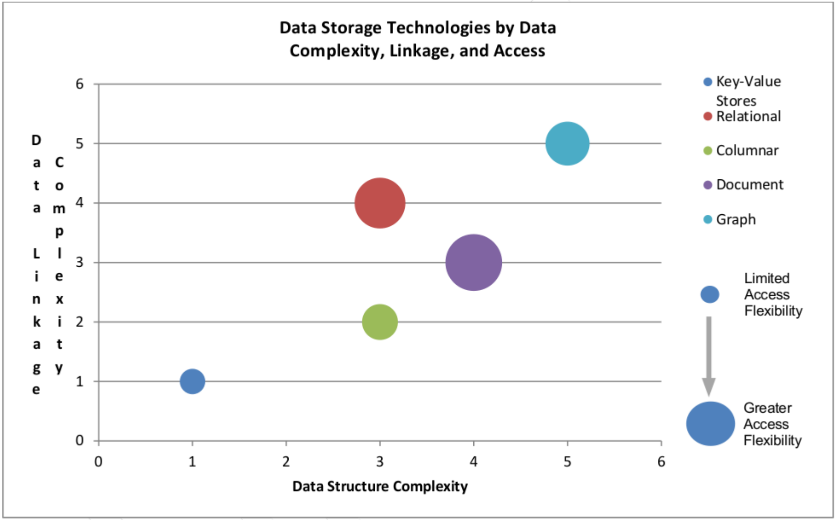


Figure 4: Data Storage Technologies

Overview: Processing

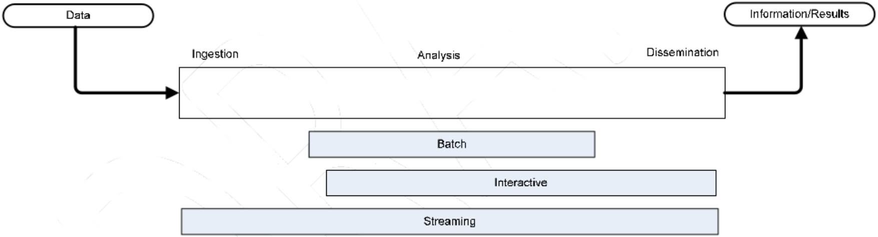


Figure 5: Information Flow

Interactive: bringing humans into the loop

Streaming: massive data streaming through system with little storage

Batch: data stored and analysed in large blocks, "batches," easier to develop and analyse

Big Data Processing: Distributed processing

breaking up computation to scale it up

Processing Background Concepts

in-memory: in RAM, *i.e.*, not going to disk

parallel processing: performing tasks in parallel

distributed computing: across multiple machines

multi-threaded processing: multiple threads on the one machine (usually shared memory)

scalability: to handle a growing amount of work; to be enlarged to accommodate growth (not just “big”)

data parallel: processing can be done independently on separate chunks of data

- **yes:** process all documents in a collection to extract names
- **no:** convert a wiring diagramme into a physical design (optimisation)

Distributed Analytics

- legacy systems provide powerful statistical tools on the desktop
  - SAS, R, Matlab

  but often-times without distributed or multi-processor support
- supporting distributed/multi-processor computation requires special redesign of algorithms
- in-database analytics systems intended to support this

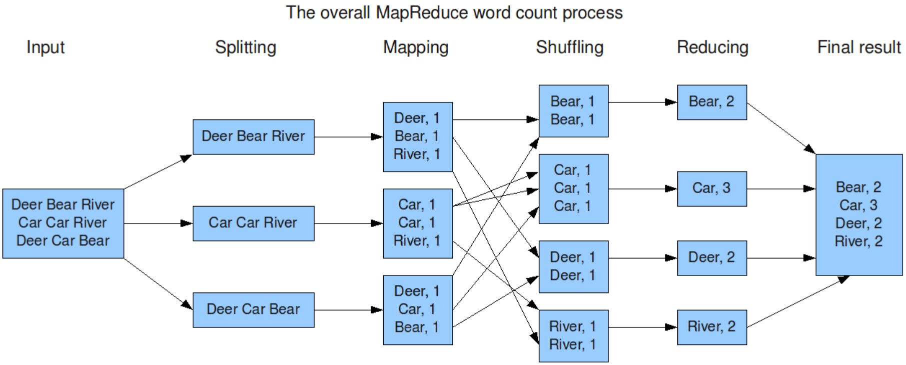
e.g. MADLib from Pivotal and MLLib from Spark integrates with their distributed SQL;

Map-Reduce

Simple distributed processing framework developed at Google

- published by Dean and Ghemawat of Google in 2004
- **intended to run on commodity hardware**; so has fault-tolerant infrastructure
- from a distributed systems perspective, is quite simple

Map-Reduce Example



for a simple word-count task: (1) divide data across machines (2) `map()` to key-value pairs (3) sort and `merge()` identical keys

Map-Reduce, cont.

- requires simple data parallelism followed by some merge ("reduce") process
- stopped using by Google probably in 2005
- Google now uses *"Cloud Dataflow"* (and *here*), available commercially, as open source

Hadoop

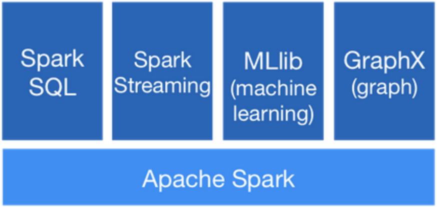
Open-source Java implementation of Map-Reduce

- originally developed by *Doug Cutting* while at Yahoo!
- architecture:

  Common: Java libraries and utilities
  YARN: job scheduling and cluster management
  HDFS: Hadoop Distributed File System
  MapReduce: core paradigm
- huge tool ecosystem
- well passed the peak of the hype curve

Spark

- another (open source) Apache top-level project at *Apache Spark*
- developed at *AMPLab* at UC Berkeley
- builds on Hadoop infrastructure (HDFS, etc.)
- interfaces in Java, Scala, Python, R
- provides in-memory analytics
- works with some of the Hadoop ecosystem



Next: Module 4
Data Resources, Processes, Standards and Tools