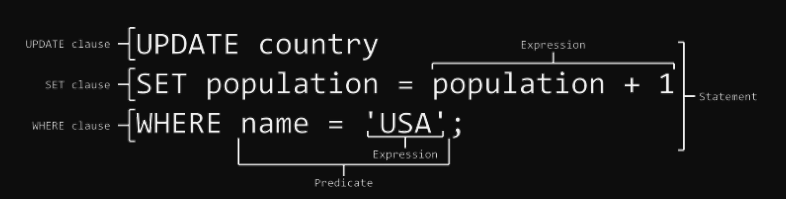
# SQL (structured query language) Review

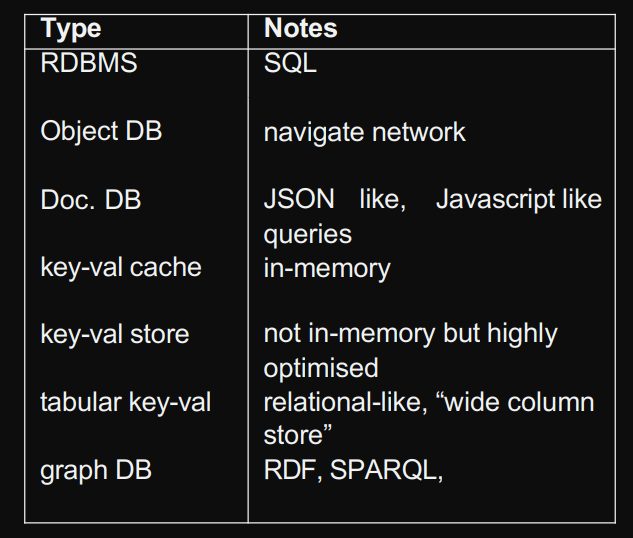


* large scale set of Excel spreadsheets with better indexing and retrieval
* Transaction oriented with support for correctness, distribution
* ACID: atomicity, consistency, isolation, and durability

## Database Background Concepts

* in-database analytics: the analytics is done within the DB
* key-value: value accessible by key, e.g., hash table
* information silo: an insular information system can’t operation with other information systems, e.g.,
  + if two big banks merge, then initially their RDBMSs will be siloed
  + insurance company, auto and home insurance customer RDBMSs may be siloed
* Many NoSQL and SQL DB offer:
  + large scale, distributed processing
  + robustness achieved
  + general query languages
  + notion of consistency

## Beyond SQL Databases

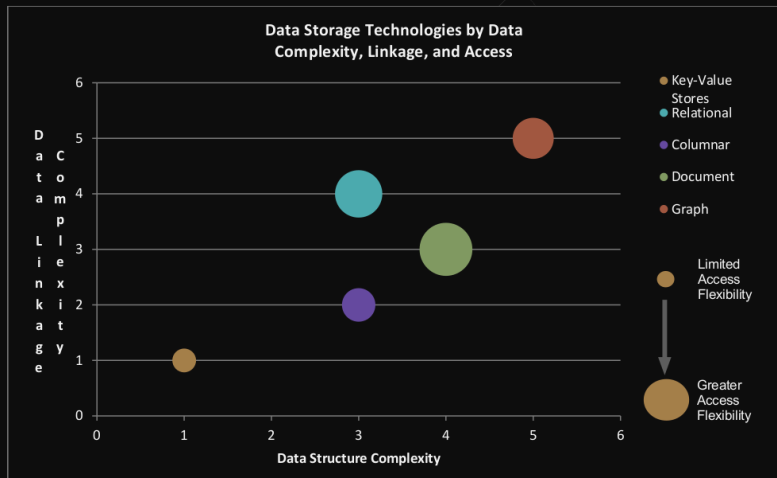


JSON and graph database is better because: doc.db, give more flexibility

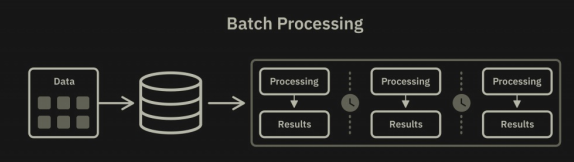
Use SQL database when data is structured and unchanging

Use NoSQL database when

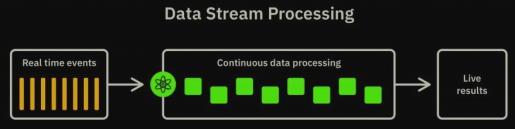
* Storing large volume of data with little to no structure
* Data changes rapidly



# Processing

Batch: data stored and analysed in large blocks 

Streaming: massive data streaming through system with little storage



Interactive: bringing humans into the loop

## Batch vs Stream

Batch:

* where you don’t need real-time analytics results
* when it is more important to process large volumes of data to get more detailed insights than speed

Streaming:

* massive data streaming through system with little storage
* Sampling can be a solution to process massive datasets

批处理一般是解决离线计算数据量大，计算时间慢的问题，流处理相反是为了解决实时计算或是近实时计算问题，当然有了实时的要求就会使处理的数据量变少，但是计算速度要求更快

## Background Concepts

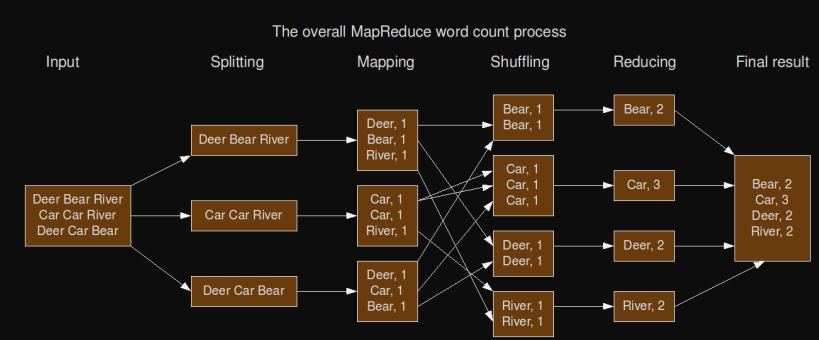
* in-memory: in RAM, i.e., not going to disk
* parallel processing: performing tasks in parallel
* distributed computing: across multiple machines
* scalability: to handle a growing amount of work; to be enlarged to accommodate growth (not just “big”)
* data parallel: processing can be done independently on separate chunks of data
  + yes: process all documents in a collection to extract names
  + no: convert a wiring diagram into a physical design (optimisation)

# Distributed Analytics

legacy systems provide powerful statistical tools on the desktop but often-times without distributed or multi-processor support

## Map-Reduce

* Simple distributed processing framework developed at Google
* To run on commodity hardware; so has fault-tolerant infrastructure
* quite simple

Example 

* for a simple word-count task:

(1) divide data across machines

(2) map() to key-value pairs

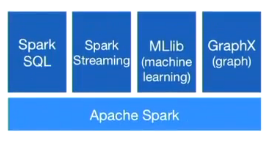
(3) sort and merge() identical keys

* requires simple data parallelism followed by reducing process
* stopped using

## Hadoop

* Open-source Java implementation of Map-Reduce
* provides an inexpensive and open-source platform for parallelising processing
* not suited to streaming (suitable for offline processing)
* This curve represents the maturity, adoption, and social application

## Spark

* interfaces in Java, Scala, Python, R
* provides in-memory analytics
* work with some Hadoop ecosystem
* 
* It is real time data processing
* includes Map-Reduce capabilities, provides real-time, in-memory processing, much faster than Hadoop