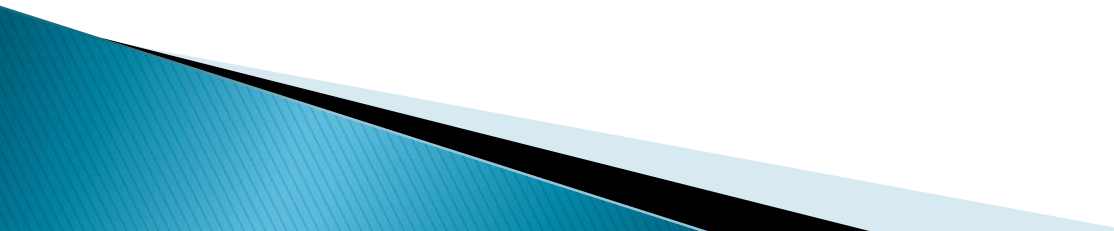


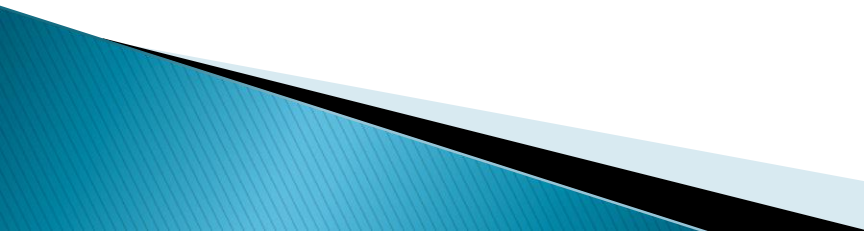
# No-SQL

**NoSQL**  
or  
**Not Only SQL**



# No-SQL

Reasons to go to a NoSQL database:

- ▶ More flexible data model (document, columnar, key-value, graph, multimodel)
  - ▶ Don't need ACID, need BASE (Basic Availability Soft-state Eventual consistency)
  - ▶ Data is distributed
  - ▶ Need horizontally scalable
  - ▶ Mostly queries, few updates
  - ▶ Asynchronous inserts & updates
  - ▶ Continuous availability
  - ▶ Big data
- 

# NoSQL

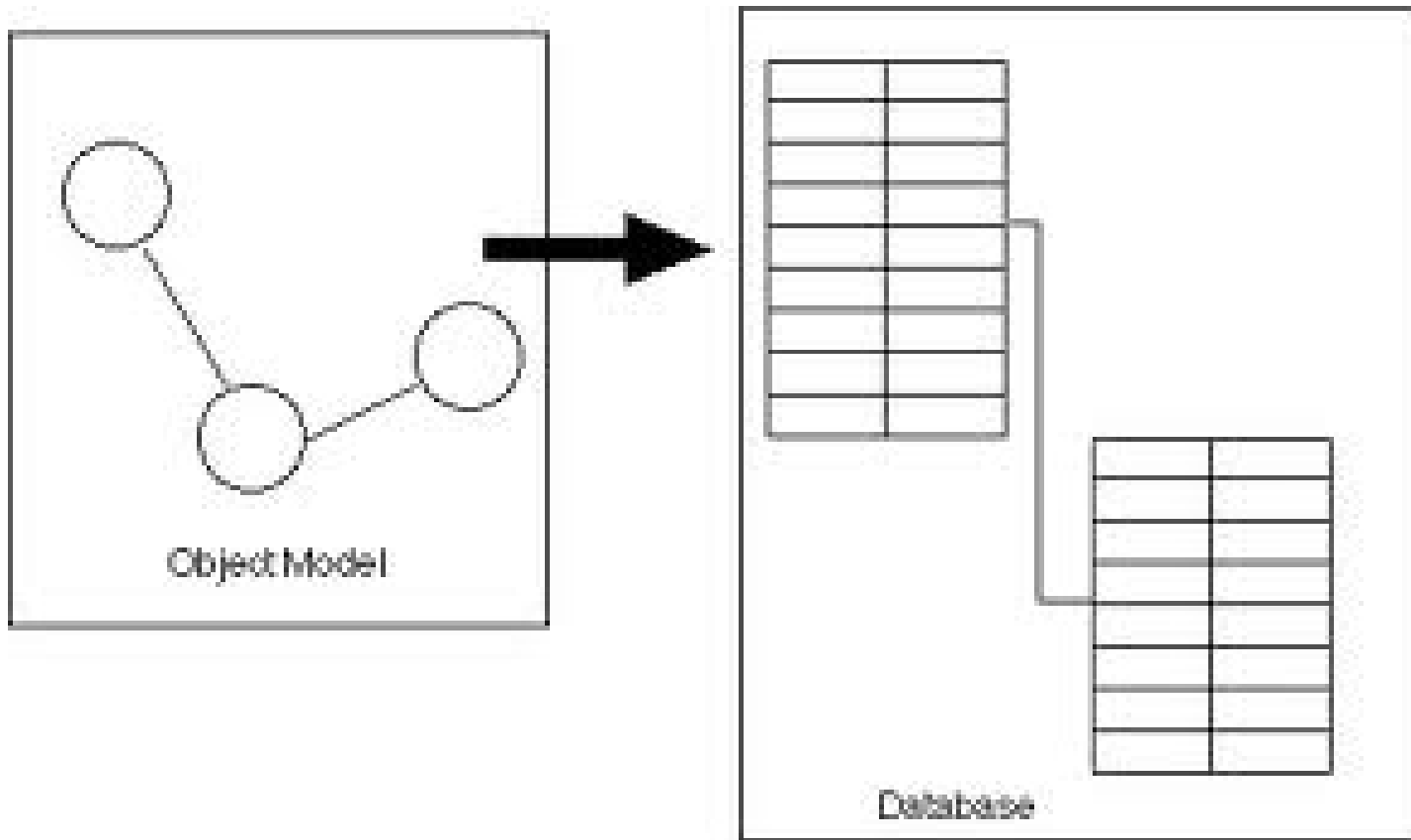
Refers to a wide variety of products

<https://gigaom.com/2012/12/20/confused-by-the-glut-of-new-databases-heres-a-map-for-you/>

# Relational Versus NoSQL

Relational DB	<u>NoSQL</u> DB
<b>Products:</b> <ul style="list-style-type: none"><li>• Oracle</li><li>• Microsoft <u>SQLServer</u></li><li>• <u>dBASE</u></li><li>• MySQL</li><li>• <u>PostgreSQL</u></li><li>• Access (toy database)</li></ul>	<b>Products:</b> <ul style="list-style-type: none"><li>• MongoDB</li><li>• Casandra</li><li>• <u>Couchbase</u></li><li>• <u>Redis</u></li><li>• Neo4j</li><li>• <u>Hadoop</u></li></ul>
<b>Designed to scale-up (vertical)</b>	<b>Designed to scale-out (horizontal)</b>
<b>Handle structured data</b>	<b>Handles semi-structured data</b>
<b>Designed for atomic transactions (ACID)</b>	<b>Eventually consistency (BASE)</b>
<b>Impedance mismatch (structure of data in DB is different than how it is stored in programming)</b>	<b>Can store and retrieve objects in more native format, like JSON documents or simple maps</b>

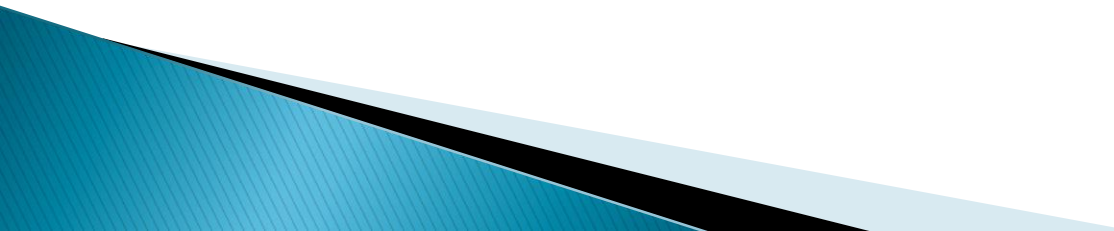
# Impedance Mismatch



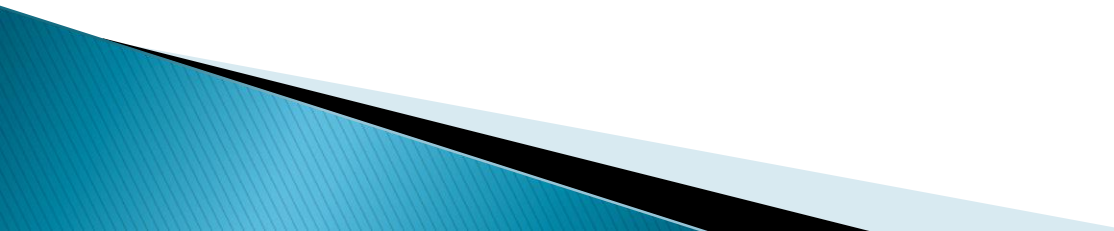
# BASE Transactions

SQL	NoSQL
<p>ACID:</p> <ul style="list-style-type: none"><li>• Atomic</li><li>• Consistent</li><li>• Isolation</li><li>• Durability</li></ul>	<p>BASE:</p> <ul style="list-style-type: none"><li>• Basically Available</li><li>• Soft state</li><li>• Eventually consistent<ul style="list-style-type: none"><li>• Stale data is ok</li><li>• Availability first</li><li>• Best effort</li><li>• Approximate answers ok</li><li>• Aggressive (optimistic)</li></ul></li></ul>

# Major Categories of NoSQL Databases

- ▶ Document-oriented
  - ▶ Columnar databases
  - ▶ Key-value stores
  - ▶ Graph stores
  - ▶ Multimodel
- 

# Document databases

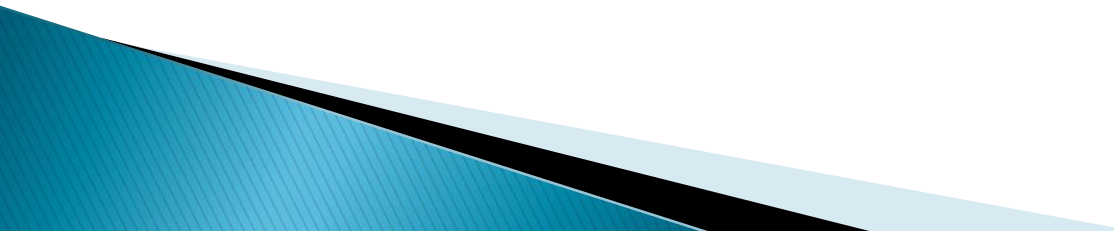
- Most popular – MongoDB and Couchbase
  - Document is a JSON file, XML file or another format
  - Good for analysis – might store browser history so that we can analyze
- 



# Columnar databases

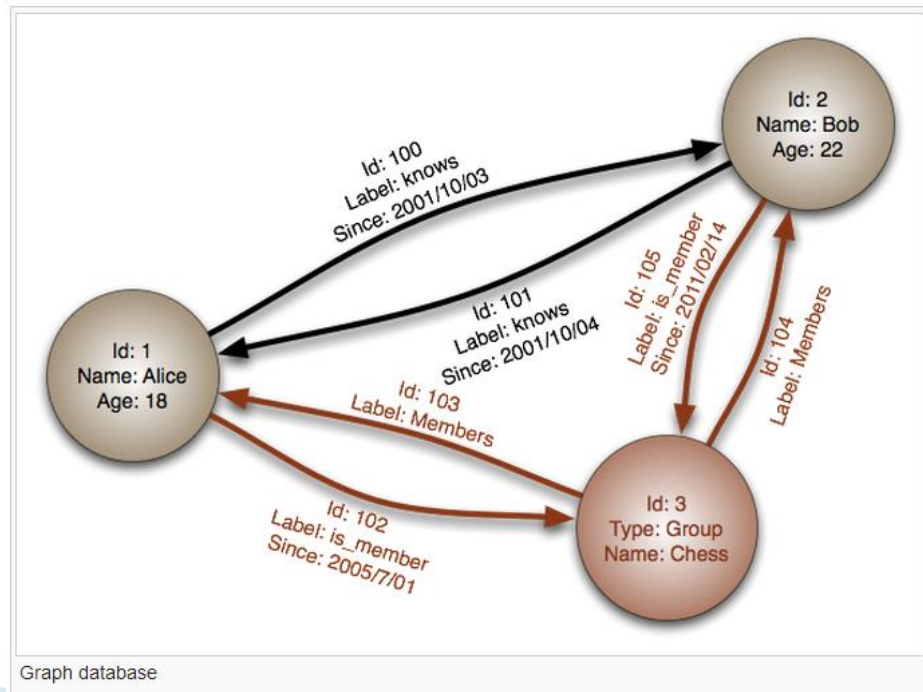
- Most popular – Casandra, DynamoDB (Amazon)
- Good for analysis – keep in this format so can slice and dice the data in different ways

# Key value database

- Most popular – Redis (used by Facebook to display your photos)
  - Like the document database except can't do a query within a document unless you have the key
  - Useful because they are really fast
- 

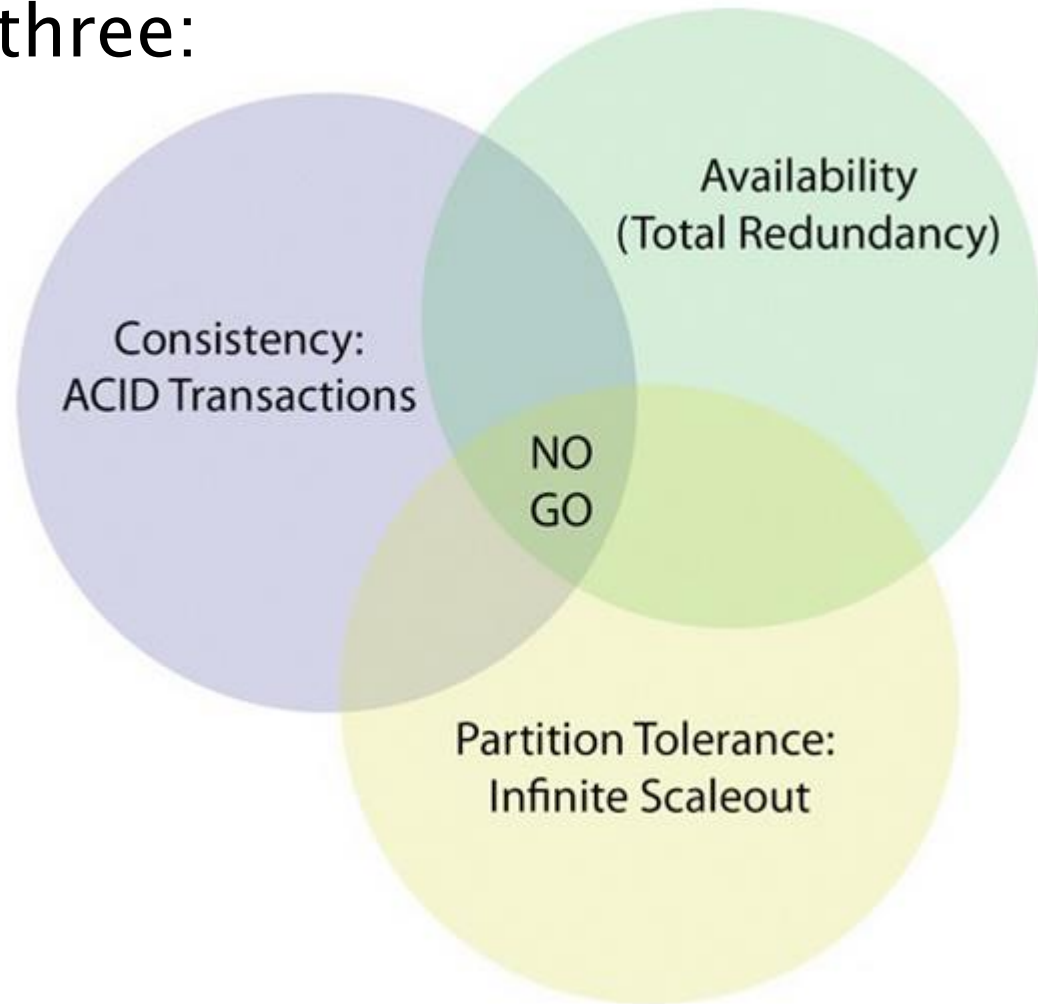
# Graph database

- Most popular – Neo4J
- Store things like – Mory is friends with George, George likes pizza, George has visited Japan, Use for things like “people who like this product are likely to also like this product”

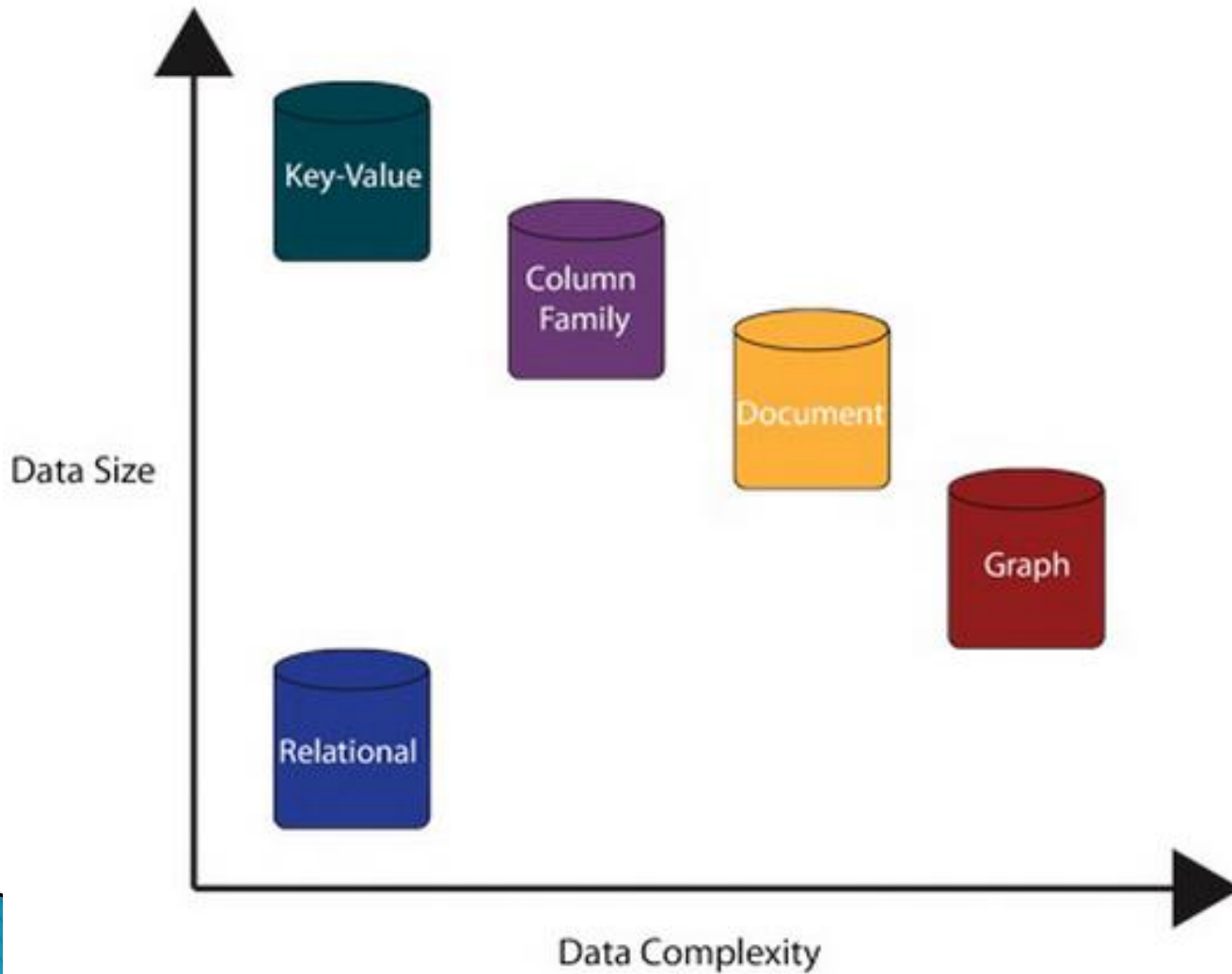


# CAP Theorem

- ▶ Can't get all three:

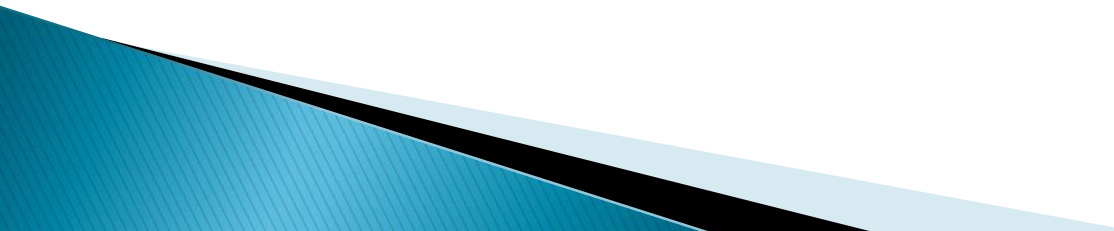


# Size versus Complexity

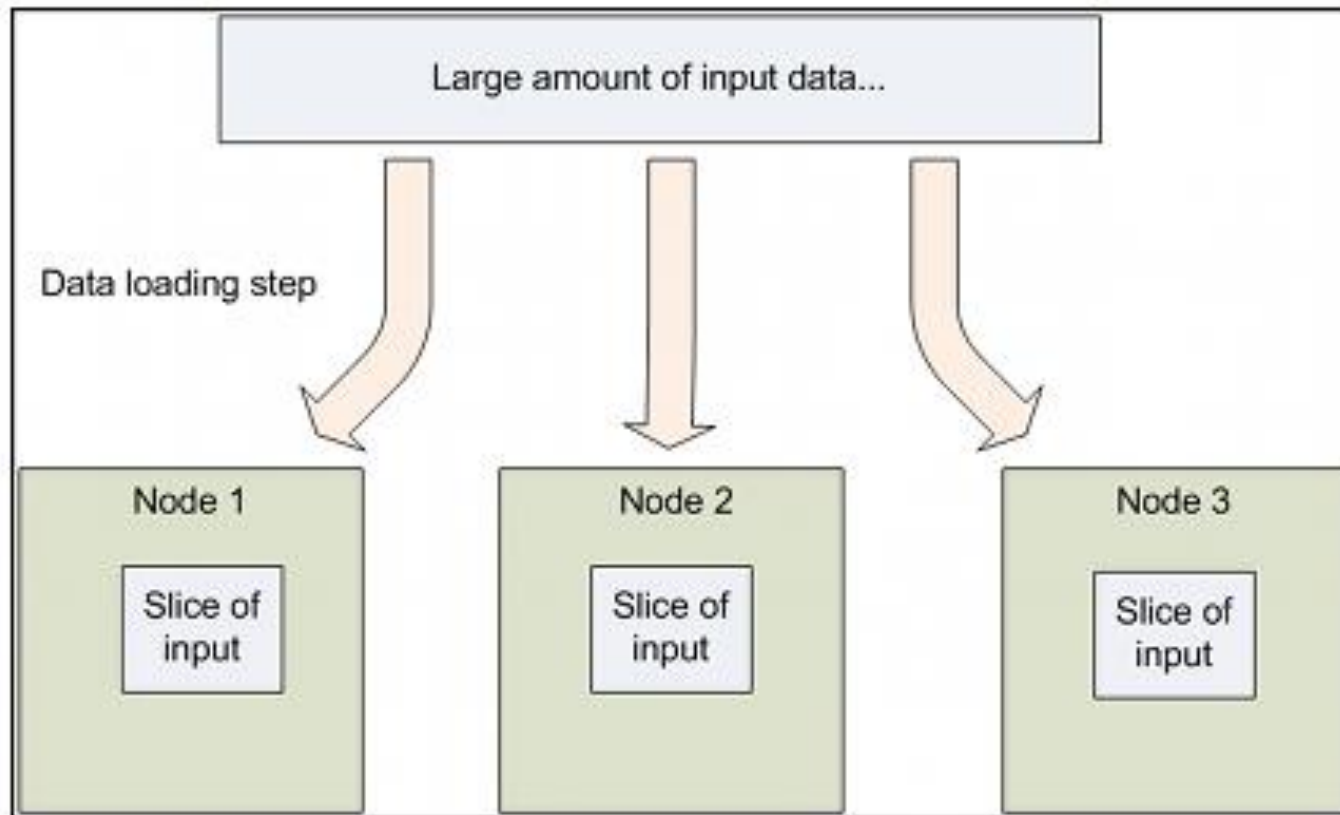


# MongoDB

## MongoDB:

- ▶ Document oriented
  - ▶ Goal – high performance, high availability, easily scalable
  - ▶ Collection (like a RDMS table) – group of MongoDB documents
  - ▶ Documents – set of key–value pairs (so schema less)
- 

# Hadoop



# Map Reduce

Map reduce – technique for indexing and searching large data volumes

Two phases, map and reduce

Map – extract sets of key–value pairs from underlying data

Reduce – merge and sorts sets of key–value pairs

