Why Hadoop, Big Data and Map Reduce.

Answer: `'because they work together really well. Each technology deals with the intrinsic problems encountered when extracting value from the data set'.`

Example technologies

I am assuming you are familiar with sql and relational databases, I will only expand the NoSQL and MapReduce examples in detail. For Mapreduce we look at Hadoop and NoSQL we look at mongodb. Mongodb is the best NoSQL database to use for a beginner in Big Data development.

Types of Big Data

Type of Big Data	Description	Prefered Technology
Structured data	Data is organised into entities that have a defined format, such as XML documents or database tables that conform to a particular predefined schema	RDBMS, NoSQL
Semi Structured data	data is looser, while there may be a schema, it can be ignored, so it may be used only as a guide to the structure of the data.	MapReduce, NoSQL
Unstructured data	Unstructured data does not have any particular internal structure, for example, plain text or image data.	MapReduce

Simple Problem 'compute the average age for the people in each city'

The Relational Database Management System (RDBMS) way

Structured Query Language (SQL)
Database people TABLE city

Id	person_id	age
1	1	20

1	2	34
3	3	17

```
SELECT people.city.id, AVG(people.city.age)
FROM     people.city
GROUP BY people.city.id
```

The Mapreduce (hadoop) way

MAP STEP (Scatter)

RECORD	KEY	VALUE
city=3, age=5	3	5
city=1, age=2	1	6
city=3, age=7	3	7
city=4, age=9	4	9
city=4, age=9	4	9
city=1, age=3	1	3

SHUFFLE STEP (Sort by Key)

KEY	VALUES
1	3, 6
3	5,7
4	9,9

REDUCE STEP (Gather)

KEY	AGGREGATION
1	(3+6)/2
3	(5+7)/2
4	(9+9)/2

The NoSQL (MongoDb) way ‘mongodb aggregation framework’

Typical Dataset object

```
{ "_id" : "35004", "city" : "Acmar", "pop" : 6055, "loc" : [-86.51557, 33.584132] }
```

aggregation function

```
puts coll.aggregate([
  {"$group" => {_id: { city: "$city"}, pop: {"$sum" => "$pop"}}},
  {"$group" => {_id: "$_id.city", avg_city_pop: {"$avg" => "$pop"}}},
  {"$sort" => {avg_city_pop: -1}},
  {"$limit" => 3}
])
```

MAPReduce Summary

Mapreduce is Parallel algorithm that follows the Scatter Gather design pattern
All map and reduce calls are done in parallel independently
MapReduce is a good fit for processes that need to analyse the whole dataset, in a batch operation, specially for ad-hoc analysis.
Mapreduce interprets the data at processing time. The input keys and values for MapReduce are not an intrinsic property of the data, but they are chosen by the person analyzing the data.
Mapreduce is linearly scalable programming model. A developers task is to write Map functions & Reducer functions piped together through a Shuffle Sort. Mapreduce defines a mapping from one set of key-value pairs to another. These function are oblivious to the size of the data or the cluster they are operating on, so they can be used unchanged for small dataset and for massive one.

RDBMS Summary

RDBMS is good for point queries or updates, where the dataset has been indexed to deliver low latency retrieval and update times of a relatively small amount of data.
MapReduce suits applications where the data is written once , and read many times, whereas a relational database is good datasets that are continually updated.
Relational data is often normalized to retain its integrity & remove redundancy.

NoSQL Summary

32-bit MongoDB only handles 2GB of data
12-node limit to the replica-set strategy with Consumption of disk space issue
In MongoDB, a **database** is composed of **collections** (somehow the equivalent of tables of those who are used to SQL). A collection contains **documents** which are JSON object
Query’s are performed on a single collection. There is no equivalent to the SQL JOIN. Hence need to ‘Denormalise’ when designing schema’s.
MongoDB doesn’t enforce a model or schema

Mongo allows optional fields where SQL doesn't
As per Mongo 2.2, there is a per-database lock, which means that operations on independent objects within the same collection can't be done concurrently.
MongoDB operations are only atomic on the document level, these locks (in traditional databases are used to guard index access) are only held as long as a single document/objects takes to update in memory.

Summary of technology attributes

Attribute	Traditional RDBMS	NoSQL	MapReduce
Data Size:	Gigabytes	Gigabytes	Petabytes
Access:	Interactive and Batch	Interactive and Batch	Batch
Updates:	Read and write many times	Read and write many times	Write once , read many times
Structure:	Static schema	Object Types	Dynamic schema
Integrity	High	High	Low
Scaling	Nonlinear	Nonlinear	Linear

Apache Hadoop Ecosystem

- Common: A set of operations & interfaces for distributed file systems & general I/O (Serialization, Java RPC, persistent data structures)
- Avro : A serialization system for efficient , cross language persistent data storage.
- MapReduce: A Distributed data processing model and execution environment thsat runs on large clusters of commodity machines.
- HDFS: A distributed filesystem that runs on large clusters of commodity machines.
- Pig: A data flow language and execution environment for exploring very large datasets. Pig runs on HDFS and MapReduce clusters.
- Hive: A distributed data warehouse. Hive Manages data stored in HDFS & provides batch style computations & ETL by HQL.
- HBase: A distributed , column oriented database, HBase uses HDFS for its underlying storage, supported both batch – style computations using MapReduce and point queries.
- Sqoop: A Tool for efficiently moving data between RDBMS & HDFS

- Mahout machine learning

Summary

1. RDBMS can scale effectively when used with a clustering technology like MySQL Cluster for example. However requires a highly structured data set. Joins can be expensive.

2. NoSQL scales effectively. For MongoDb there are some architectural limitations that could for massive data sets be a bottleneck for capacity. Elastic schemas, however no joins hence need to ‘de-normalise’.

3. MapReduce, scales to largest data set. Can optimise writes however reads are slow and as dataset is distributed need to stream result set for queries.