CSCU9YQ - NoSQL Databases
Lecture 1: Introduction

Gabriela Ochoa

http://www.cs.stir.ac.uk/~goc/

Admin

- Lectures
  - Tuesdays 15:00 & 16:00 (10 min. break), LTA5
  - 10 Lectures: January 15, 22, 29. February 5 & 12
- Labs (attendance, no checkpoints)
  - Tuesdays, at 10:00 & 12:00, 4X5 (self-sign in Canvas)
  - 5 labs: January: 22, 29. February: 5, 12 & 26
- Evaluation
  - 50% assignment on MongoDB
    - Both written analysis & technical implementation sections
    - Deadline:
  - 50% exam

Relational Databases

itemid	orderid	item	amount
5	1	Chair	200.00
6	1	Table	200.00
7	1	Lamp	123.12

customerid	name	email
5	Rosalyn Rivera	rosalyn@adatum.com
6	Jayne Sargent	jayne@contoso.com
7	Dean Luong	dean@contoso.com

orderid	customerid	date	amount
1	4	11/1/17	523.12
2	3	11/15/17	32.99
3	1	11/21/17	23.99

Relational Model

- Data expressed as *tuples* (a set of attribute/value pairs.) -> Rows
- A set of tuples that all share the same attributes is called a *relation* -> Table
- *Schema*: table, column names &types. Stable over time. Not expected to change
- Primary keys uniquely identify rows within a table
- Foreign key fields are used in one table to refer to a row in another table
- Uses the Structured Query Language (SQL)

Relational Databases

- Dominant technology for over 20 years!
- Why is it so successful? Because it provides
  - Persistence: keep large amounts of persistent data
  - Concurrency control: many users simultaneously. Coordinate to avoid errors. This is done via *Transactions*
  - Integration mechanisms: multiple applications, written by different teams, access the same data.
  - A standard model: developers can learn the relational model and apply it in many projects. SQL dialects are very similar

What is a Data Model?

- In general, a model is a perception of the structures of reality (system we want to model)
- Data Models contain formalisms for expressing
  - Data structures
  - Constraints
  - Operations
  - Keys and identifiers
  - Integrity and consistency
- Describes how we interact with data in the DB

What is NoSQL?

- New generation of databases that differ from the Relational Model
- New features and practices that are best suited for a new type of applications and “Big Data”
- RDBs excel at maintaining consistency, but many sacrifice performance (schema checks)
- There are limits to how big RDBs can scale
- NoSQL DBs focus on performance over consistency
  - Data have structure, but without enforcing fixed schema
  - Data is replicated across many nodes asynchronously

What is Big Data?

Big Data - Massive amounts of complex data that require special techniques for acquisition, storage, distribution and analysis.

- any aspects of our lives produce data: Shopping, communicating, reading news, music, searches, expressing opinions – all is tracked!
- Transmitted by sensors and mobile devices
- Traditional data processing software is inadequate
- Much of this data comes in an unstructured form (i.e. not structured tables in row & columns)

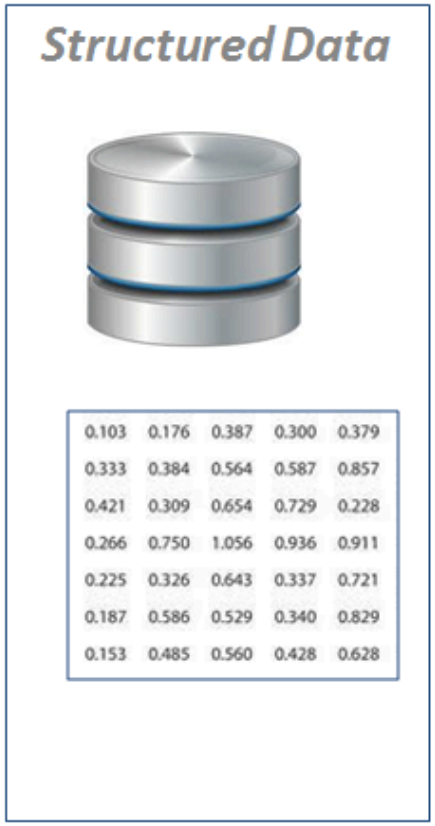
Big Data

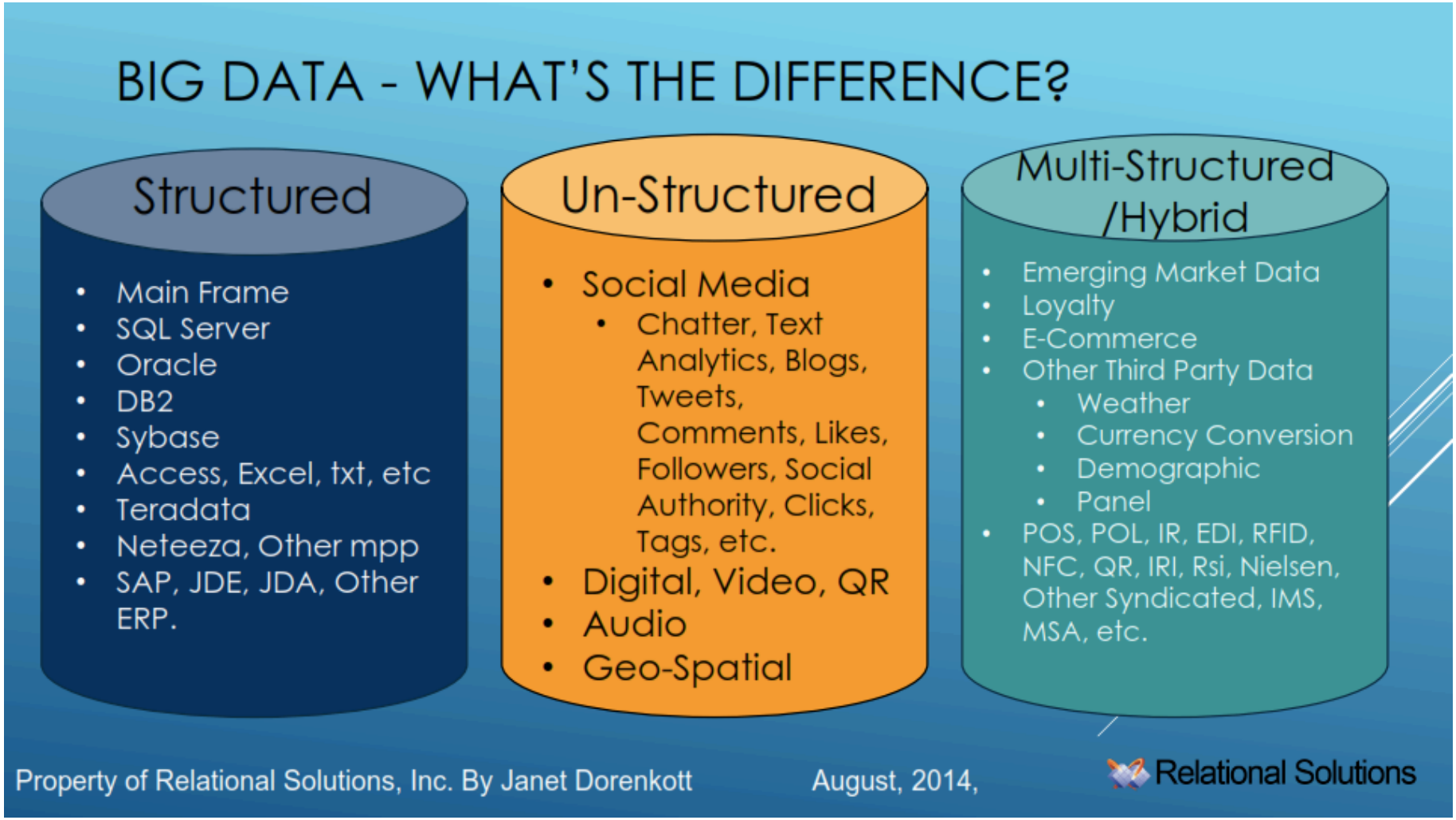
Structured

- Relational databases
- Spread sheet

Unstructured

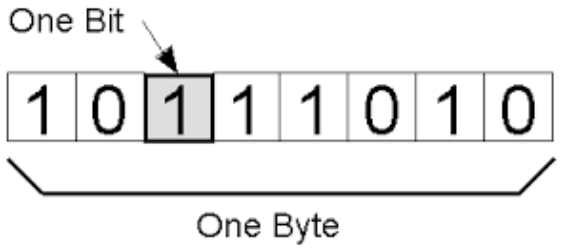
- Text and multimedia content
- e-mail, videos, audio, written documents
- Geospatial data





There are structured, unstructured and hybrid forms of data.
All are relevant for Big Data

Units of data storage

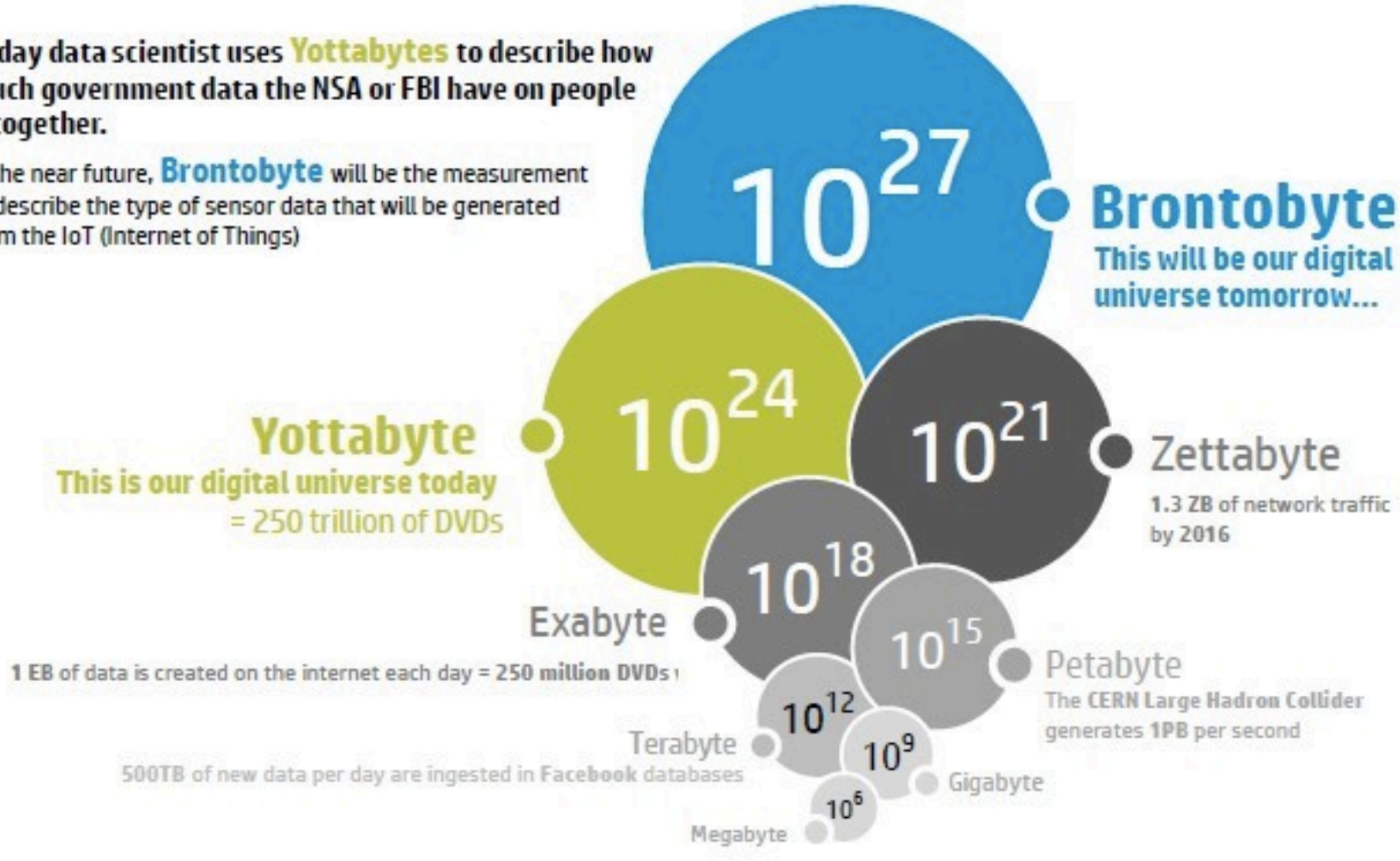


Unit	Value	Size
bit (b)	0 or 1	1/8 of a byte
byte (B)	8 bits	1 byte
kilobyte (KB)	1000^1 bytes	1,000 bytes
megabyte (MB)	1000^2 bytes	1,000,000 bytes
gigabyte (GB)	1000^3 bytes	1,000,000,000 bytes
terabyte (TB)	1000^4 bytes	1,000,000,000,000 bytes
petabyte (PB)	1000^5 bytes	1,000,000,000,000,000 bytes
exabyte (EB)	1000^6 bytes	1,000,000,000,000,000,000 bytes
zettabyte (ZB)	1000^7 bytes	1,000,000,000,000,000,000,000 bytes
yottabyte (YB)	1000^8 bytes	1,000,000,000,000,000,000,000,000 bytes

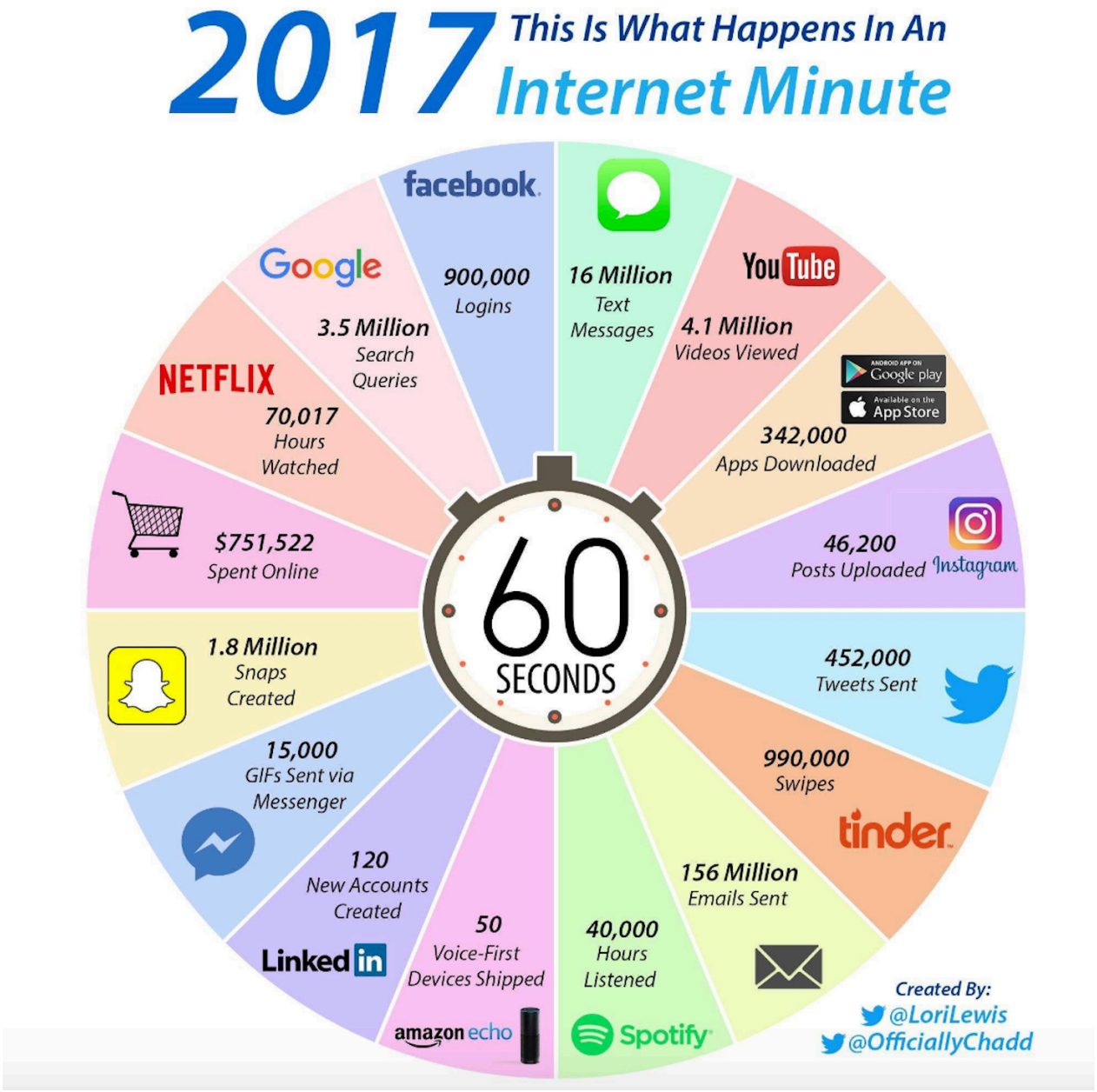
Brontobytes …

Today data scientist uses Yottabytes to describe how much government data the NSA or FBI have on people altogether.

In the near future, **Brontobyte** will be the measurement to describe the type of sensor data that will be generated from the IoT (Internet of Things)



https://www.theregister.co.uk/2012/12/04/hp_discover_autonomy_vertica_big_data/



- The incredible scale of e-commerce, social media, email, and other content creation that happens on the web.
- Created each year by Lori Lewis and Chadd Callahan of Cumulus Media
- http://www.visualcapitalist.com/happens-internet-minute-2017/

Main Factor for No-SQL Emergence: Clusters

- Increase of scale (what is now called Big Data), produced a need of more computing resources
- Two options
  - Scaling up (vertical): bigger machines, more processors, disk, storage and memory. More expensive! (also limits)
  - Scaling out (horizontal): use a lots of small machines in a cluster. Much cheaper, and more resilient (keep going despite failures)
- RDB are not designed to run efficiently on clusters, while NoSQL have been designed to run on clusters

Other Factors

- Object Relational Impedance Mismatch
  - We like to program in object oriented languages
  - The objects don't match well with the structure of the RDB
- From Integration DB to Application DB
  - Movement away from using DB as integration – multiple applications, several teams
  - Towards an Application DB – only looked by a single application, single team. Application responsible for DB integrity.

Application Databases

- *Service Oriented Architectures* (web services) communicate over HTTP in a format that is divorced from the database
- e.g. XML or JSON, richer data structures
- NoSQL DB offer more flexibility for the data structure to communicate
- Developers can choose the right DB for the right application

The NoSQL Term

- Term coined in 2009 by Johan Oskarrson who needed a short hashtag for a meeting he was arranging (accidental neologism)
- Not a good name for a number of reasons
  - Says what it is not, not what it is
  - Best used as an umbrella term for new generation databases
  - There is no prescriptive definition. Best described as a set of common characteristics

Characteristics of NoSQL Databases

- Not using the Relational Model
- Running well on clusters
- Open-source
- Flexible schemas (freely add fields to DB records, without needing to define a fixed schema first)
- Big data, web applications

Types of NoSQL DB by Data Model

Data Model	Example Databases
Key-Value	BerkeleyDB, LevelDB, Memcached, Project Voldemort, Redis, Riak
Document	CouchDB, *MongoDB*, OrientDB, RavenDB, Terrastore
Column- Family	Amazon SimpleDB, *Cassandra*, Hbase, Hypertable
Graph	FlockDBm HyperGraphDB, Infinite Graph, *Neo4J*, Orient DB

- This course will cover: MongoDB, Cassandra & Neo4j.
- Emphasis on MongoDB (Labs, Assignment)

Aggregate Data Models

- From the 4 types of NoSQL DB, the first 3 (Key-value, Document and Column-family) share an *Aggregate* orientation
- Aggregate orientation
  - Recognise the need to operate in units that have a more complex structure than a set of rows
  - Such as a complex record with lists and other records nested
- Aggregate: a collection of related objects that we whish to treat as a unit (for manipulation & consistency)

Example: Product Review Database

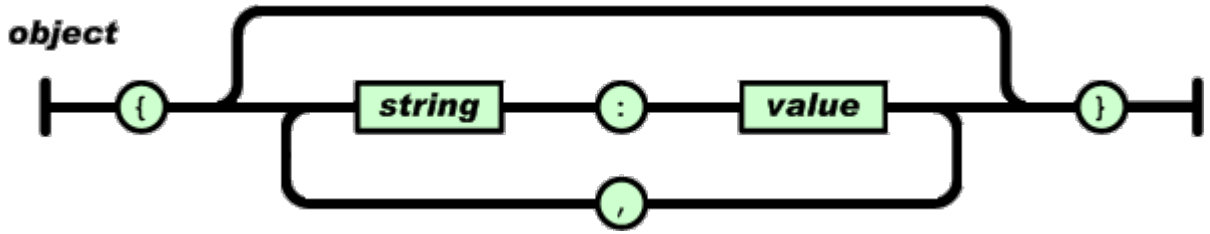
Product: iPhone 5
Price: £500
Camera: Fine
Screen: Very good
Accessories: Headphone, Case, ...

Product: iPhone 5
Price: £500
Camera: Excellent
Screen: Poor in sunshine
Operating system: Easy to use

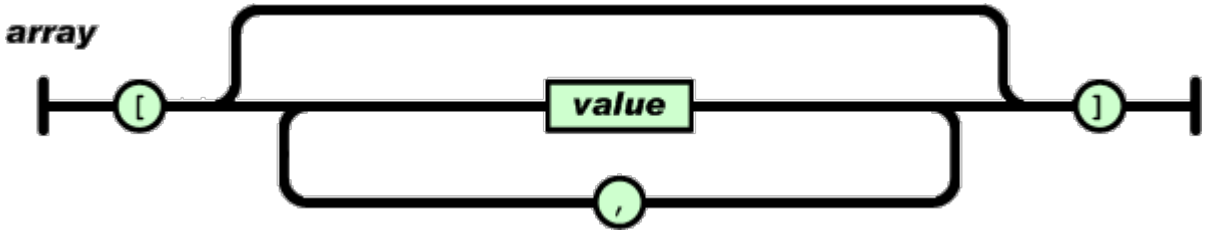
- Different products have different fields
- Some fields have several values
- Products are related to other products as accessories
- A new product may have new qualities (fields) not yet in the database
- The same product may appear many times with different values and fields

JSON: JavaScript Object Notation

- Light-weight text data interchange format, language independent
- Easy for humans to read and write
- Easy for computers to parse and generate



A collection of **name/value pairs** (object, record, struct, dictionary, hash table, keyed list, assoc. array)



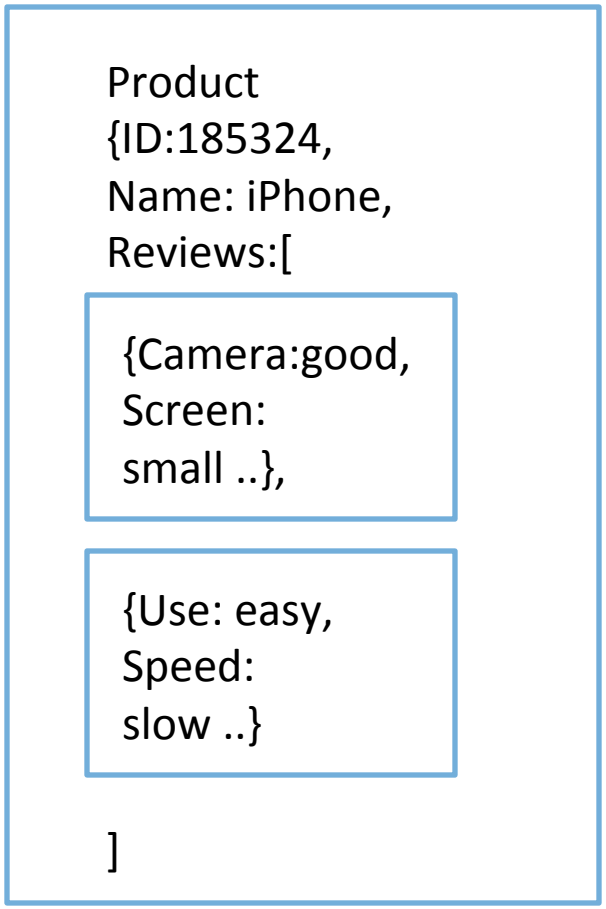
An **ordered list** of values (array, vector, list, sequence)

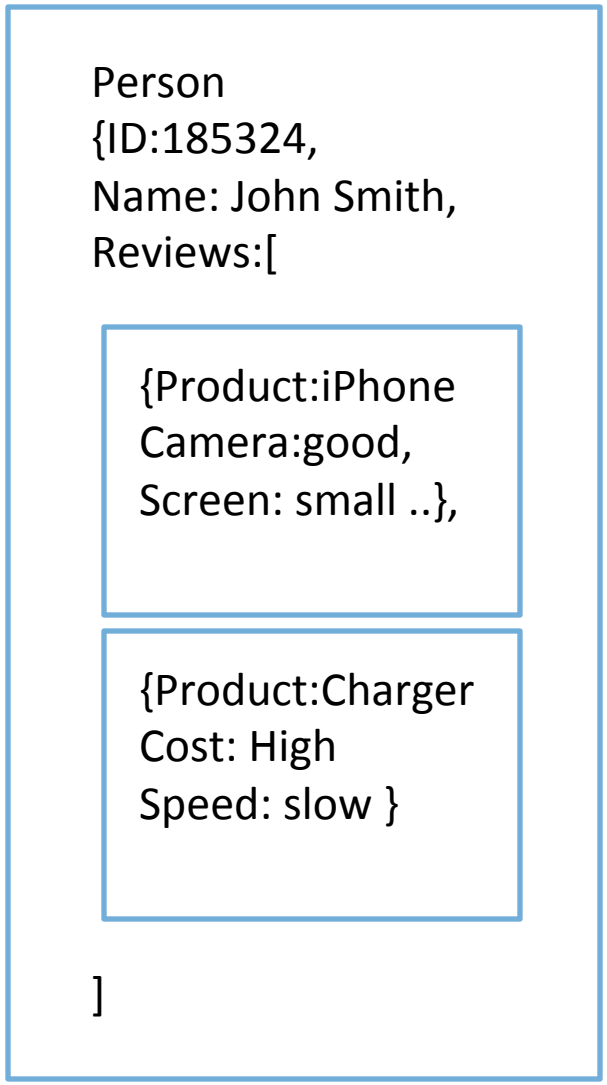
Example Review in JSON

```
{
Id:847543,
Name:iPhone5,
Features:[GPS,Retina Display,Siri],
Reviews:[
{Reviewer:458743,Date:12.4.1013,Speed:Slow},
{Reviewer:636534,Date:2.5.1013,Camera:Great},
]
}
```

Aggregate Design

- There is a new freedom, away from ER models
- But with more choice come more decisions...
- For our product review database, potential units to aggregate are:
  - Users – keep all a person's reviews together
  - Products – keep all the reviews of a product together


Product
{ID:185324,
Name: iPhone,
Reviews:[
{Camera:good,
Screen:
small ..},
{Use: easy,
Speed:
slow ..}
]


Person
{ID:185324,
Name: John Smith,
Reviews:[
{Product:iPhone
Camera:good,
Screen: small ..},
{Product:Charger
Cost: High
Speed: slow }
]

What is the best aggregate?

- Which design you choose depends in part on what is the most common query?
  - List all reviews from a given customer
  - List all reviews of a given product
  - Find all products that are ‘fast’

Aggregates help with NoSQL ACID

- Atomic, Consistent, Isolated and Durable: Relational databases key strength and requirement for transactions
- Aggregates help impose atomicity in NoSQL DB – an aggregate is updated in a single transaction
- Aggregates are central to to running on a cluster – data for an aggregate is stored on one node
- NoSQL types (key-value, Document, Colum) share the notion of an aggregate indexed by a key. They differ in the characteristics of the aggregate.

Summary

- Data Models (Relational and Others)
- What is NoSQL?
- What motivated the emergence of NoSQL?
- Types of NoSQL DBs
- The Aggregate Data Model