

SCHOOL OF COMPUTER AND COMMUNICATION SCIENCES

---

## Applied Data Analysis Summary

---



Dr. CATASTA Michele  
Distributed Information Systems Laboratory (LSIR)  
[michele.catasta@epfl.ch](mailto:michele.catasta@epfl.ch)

June 10, 2016

## Contents

|          |                                                                         |           |
|----------|-------------------------------------------------------------------------|-----------|
| <b>1</b> | <b>Introduction</b>                                                     | <b>3</b>  |
| 1.1      | General information about the course . . . . .                          | 3         |
| 1.2      | Data Science . . . . .                                                  | 3         |
| <b>2</b> | <b>Basic concepts</b>                                                   | <b>5</b>  |
| 2.1      | Panda vs SQL . . . . .                                                  | 6         |
| 2.2      | OnLine Analytical Processing (OLAP cubes) . . . . .                     | 6         |
| <b>3</b> | <b>Data Wrangling</b>                                                   | <b>8</b>  |
| 3.1      | Diagnosis of the data . . . . .                                         | 9         |
| 3.2      | Dealing with missing values . . . . .                                   | 9         |
| 3.3      | General procedure . . . . .                                             | 10        |
| <b>4</b> | <b>Data Variety</b>                                                     | <b>10</b> |
| 4.1      | Role of Schema . . . . .                                                | 11        |
| 4.2      | Examples of data . . . . .                                              | 11        |
| 4.2.1    | XML and DOM . . . . .                                                   | 11        |
| 4.2.2    | JSON . . . . .                                                          | 12        |
| 4.2.3    | Tabular data . . . . .                                                  | 12        |
| 4.2.4    | Log files . . . . .                                                     | 13        |
| 4.2.5    | Binary formats . . . . .                                                | 13        |
| 4.3      | Processing the data (JSON and XML) . . . . .                            | 13        |
| 4.4      | HTML and Web Services . . . . .                                         | 14        |
| 4.4.1    | HTML . . . . .                                                          | 14        |
| 4.4.2    | Web Services . . . . .                                                  | 14        |
| <b>5</b> | <b>Statistics on the Data</b>                                           | <b>16</b> |
| 5.1      | Example of famous mistakes due to statistics . . . . .                  | 16        |
| 5.1.1    | Anscombe's quartet: Sensivity of outliers & Robust statistics . . . . . | 16        |
| 5.1.2    | Simpson's paradox: aggregation of data . . . . .                        | 16        |
| 5.2      | Refresh of basic stats . . . . .                                        | 17        |
| 5.2.1    | Bayes Theorem . . . . .                                                 | 17        |
| 5.2.2    | Random Variables . . . . .                                              | 18        |
| 5.2.3    | Law of Large Numbers . . . . .                                          | 18        |
| 5.3      | Measurement on Samples . . . . .                                        | 18        |

# 1 Introduction

## 1.1 General information about the course

This course covers multiple topics in the data science field such as **Data Wrangling**, **Data Management**, **Data Mining**, **Machine Learning**, **Visualization**, **Statistics** and **Story telling**. It's about **breadth**, not depth. Indeed, Data science is evolving really quickly, hence learning in depth a specific tool won't pay off.

## 1.2 Data Science

When we talk about Data Science, we often use the term Big Data as the enormous amount of data that exist in the world. But Big Data is not only about collecting huge amount of data. It is challenging but not enough. The real value comes from the insights. The *internet* companies (Google, Facebook, etc.) understood this many years ago.

An accurate definition of Data Analysis is given by Wikipedia:

*Analysis of data is a process of inspecting, cleaning, transforming, and modeling data with the goal of discovering useful information, suggesting conclusions, and supporting decision-making. Data analysis has multiple facets and approaches, encompassing diverse techniques under a variety of names, in different business, science, and social science domains.*

[Wikipedia - Data Analysis](#)

Therefore, a Data Scientist has to master different kind of skills such as **Mathematics** (for the Statistics), **Programming** and the **Domain Expertise**. Drew Conway's Venn diagram, Figure 1, shows the different combination man can obtain with these three skills.

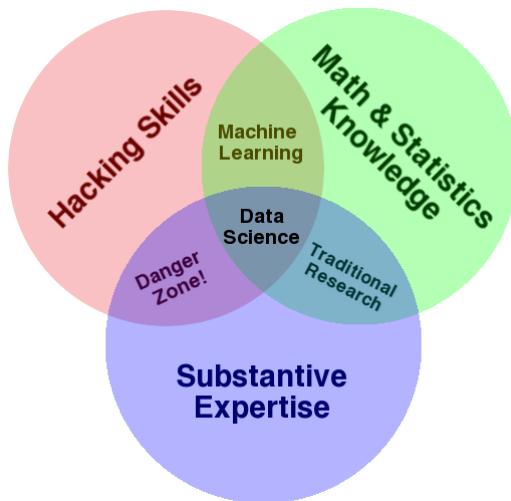


Figure 1: Venn Diagram describing the different combination of skills used by a Data Scientist (by Drew Conway)

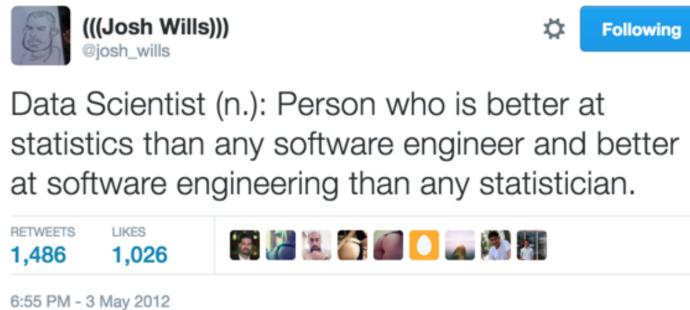
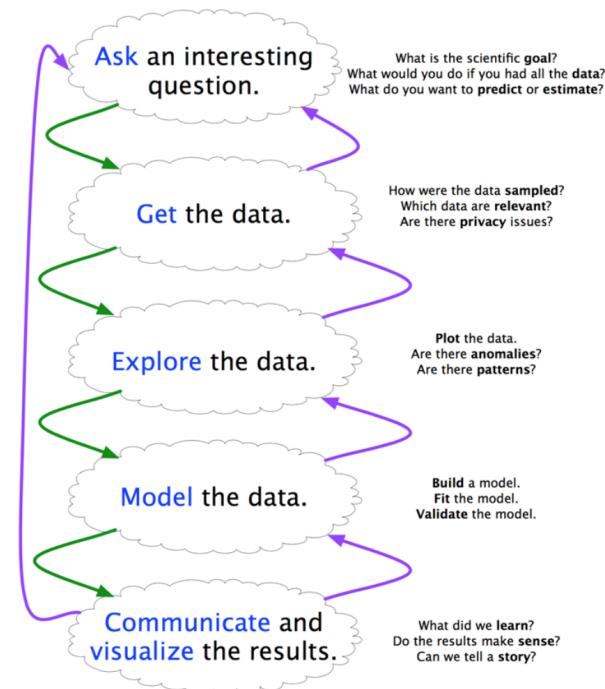


Figure 2: A tweet from Josh Wills, Data Scientist at Slack.

### A practical definition of Data Science

Data Science is about the whole processing pipeline to extract information out of data. As such, a Data Scientist **understands and cares about the whole data pipeline**.



A **data pipeline** consists of 3 steps:

1. Preparing to run a model.  
*Gathering, cleaning, integrating, restructuring, transforming, loading, filtering, deleting, combining, merging, verifying, extracting, shaping*
2. Running the model
3. Communicating the results

A “good” Data Scientist will always go back and forth between the steps. The diagram on the left shows exactly what can happen.

In this course, you will develop the following skills:

**data muning/scraping/sampling/cleaning** in order to get an informative, manageable dataset

**data storage and management** in order to be able to access data quickly and reliably during subsequent analysis

**exploratory data analysis** to generate hypotheses and intuition about the data

**prediction** based on statistical tools such as regression, classification, and clustering

**communication of results** through visualization, stories and interpretable summaries

## 2 Basic concepts

A data science student is attended to understand the **Grammar of Data Science**. Having some backgrounds in SQL concepts is also a good thing because, as it is very common, people loves to make example with it. Here is a brief refresh of some definitions and concepts about data science.

- **Structured data** requires two key concepts:
  - **Data model** is a collection of concepts for describing data.
  - **Schema** is a description of a particular collection of data, using a given data model.
- The **Relational model** is one of the most common approach to manage data (SQL like) and can handle most of the data. A counter example is the facebook-like data which requires **graph model**. This model is made of 2 parts:
  - The **Schema**.  
For example, `Students(sid: string, name:string, age:integer)`
  - The **Instance**, *i.e.* the data at a given time.  
Definitions:
    - \* **Cardinality** is the number of rows. (Number of items)
    - \* **Degree or Arity** is the number of fields. (Number of attributes)
- Definitions of some “Database” terms:
  - A **JOIN** is a mean to combine tables based on shared attributes (most of the time some **IDs**). Despite its apparent simplicity beware of the many ways to compute a JOIN and check what is the default JOIN of a language before using it. The FIG 3 summarises these possibilities.
  - **Aggregation, reduction**, and **groupby** are the action of reducing data with a common operation (**sum, count, average, ...**) to summarise them.
- Definitions of some “Pandas” terms:
  - **Series** are a name, ordered dictionary
    - \* keys are indexes
    - \* built on **numpy.ndarray** (so values can be any Numpy data type)
  - **DataFrame** is a table with named column
    - \* the columns are series
    - \* it is indeed a dictionary with (columnName → series)

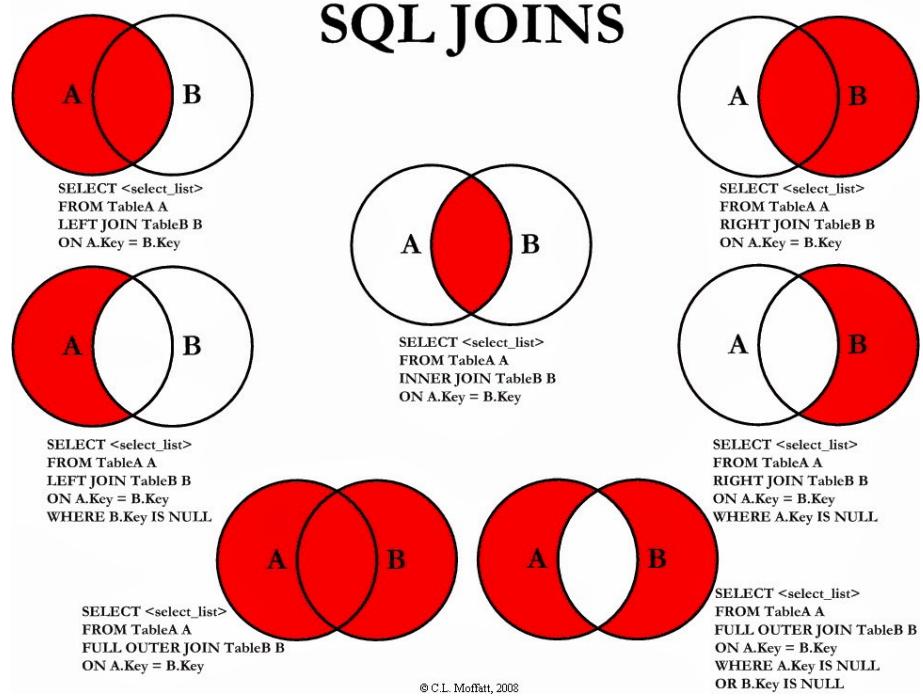


Figure 3: Different ways to join two tables and the related SQL command.

## 2.1 Panda vs SQL

Panda is built to allow easy and fast **data exploration** and not to be a database manager, as SQL is. Thus there are benefits and drawbacks of using it.

| Pros                                                                                                                  | Cons                                                                                                                                       |
|-----------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Lightweight & fast<br>Great expressiveness (combine SQL + Python)<br>Easy plot for data visualization (eg Matplotlib) | Tables stored directly in memory<br>No post-load indexing functionality<br>No transactions, journalings<br>Large, complex joins are slower |

## 2.2 OnLine Analytical Processing (OLAP cubes)

OLAP tools enable users to analyze multidimensional data interactively from multiple perspectives. Conceptually, it is like an n-dimensional spreadsheet (a cube) on which we can apply various operations to take decisions.

OLAP cubes are another way to see data table and are constructed based on them, as shown FIG 4.

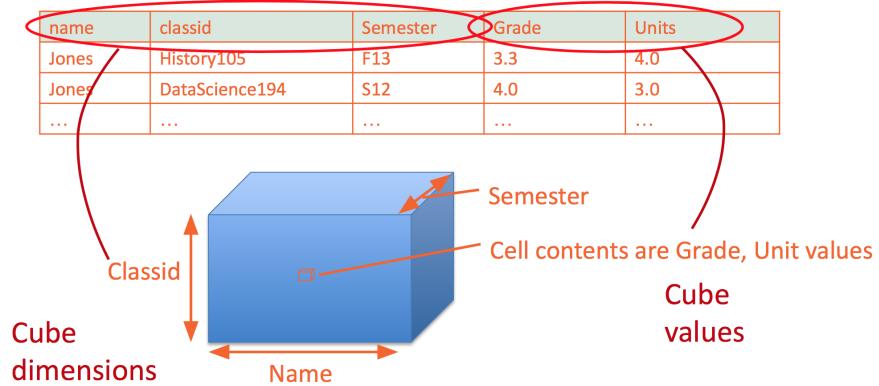


Figure 4: Construction of an OLAP cube from a table.

Operations on OLAP cubes are the following and are illustrated on FIG 5

- **Slicing** fixes one or more variable
- **Dicing** selects a range of one or more variable
- **Driling up/down** changes levels of a hierarchically-indexed variable, ie "zoom" on a variable and see the sub-categories it contains.
- **Pivoting** change the point of view of the cube. Swap an aggregated variable an a detailed one.

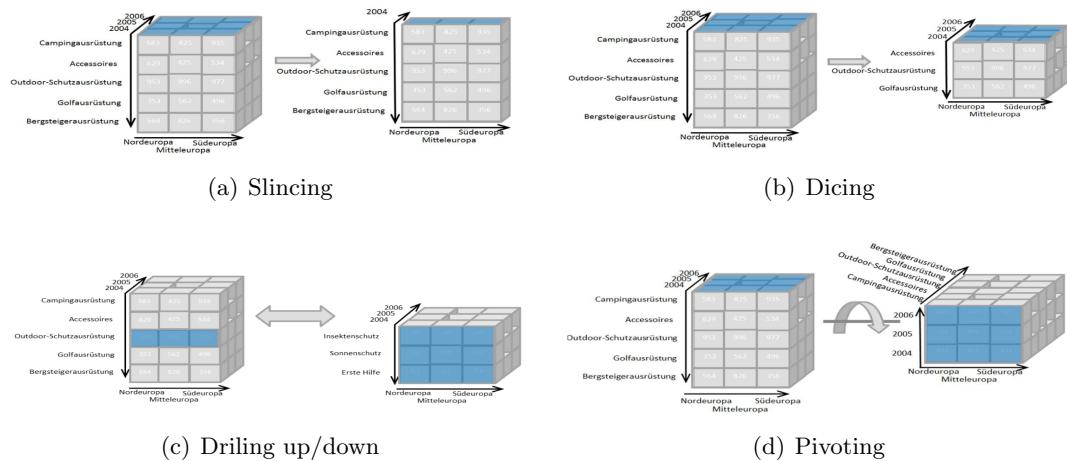


Figure 5: Operations on OLAP cubes

| Pros                                                                                                                                                                                                                                                                                    | Cons                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| The main advantage of OLAP cubes is that they are <b>conceptually simpler</b> to understand by a non-scientist person, eg a business man who have to take day-to-day decisions based on company's data. Aggregations are limited but cover the main common cases that we can encounter. | Because of the "on-line" behaviour of this approach, all type of aggregation must be pre-calculated among all combination of axis which is very <b>expensive in memory and in time</b> (when updating the data) |

### 3 Data Wrangling

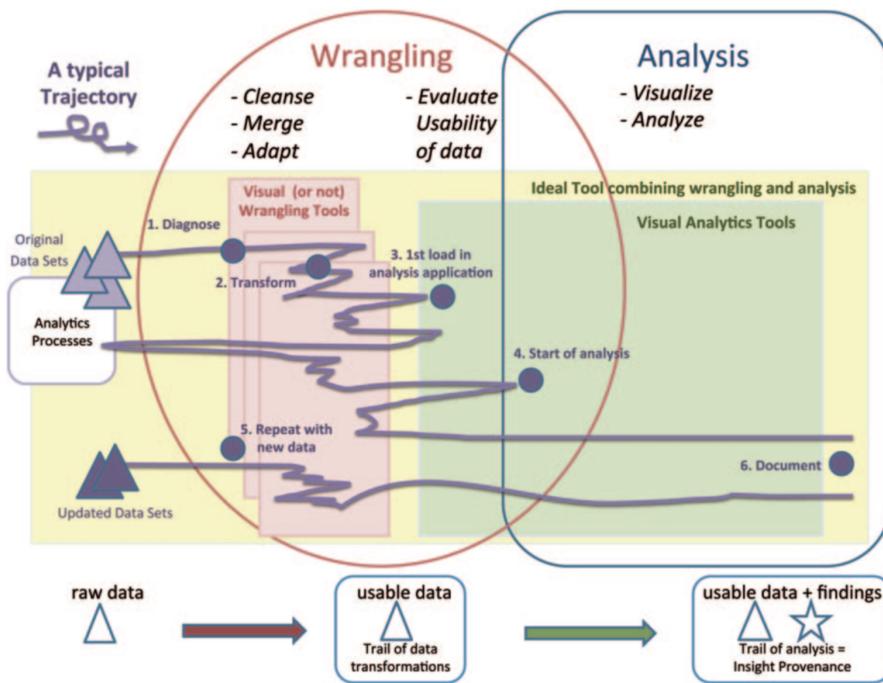


Figure 6: Things do not always happen as expected...

Before any analysis, data need to be transformed from "dirty" to clean and processable data.

Data comes from different sources (excel or SQL?), sometime collected through different methods over time, with different conventions (space or NaN?), etc ... Data wrangling's goal is to **extract and standardize these raw data**. The best way to do it is to **combine automation with visualizations** in order to find outliers.

Data's problem can come from (non-exhaustive):

- Missing data
- Incorrect data
- Inconsistent representations of the same data

- Non-standardized data (centimeter or inches? farenheit or celsuis ?)
- Duplicated data

About 75% of theses problem will need **human intervention** to be corrected (by the data-scientist or by crowdsourcing).

Even if it seems really dirty, **beware not to over-sanitize the data!**. Applying what we can call "defensive programming" is not a good idea because we risk to lose any interesting data, keeping only the ones that fit perfectly in our model.

### 3.1 Diagnosis of the data

One of the most important aspect of Data Wrangling is to **understand** the data and to **find possible problems**. In order to "diagnose" the data, two tools can be used:

- **Visualization** (A *toughful* visualization will always help)
- **Basic Statistics**

Matrix visualizations of the facebook graph is shown in Figure 7. The Relational visualization, Figure 7(a), does not show any particular problem in the data. But the Time dependant visualization, Figure 7(b), shows that the Facebook API reached its limit while collecting data.

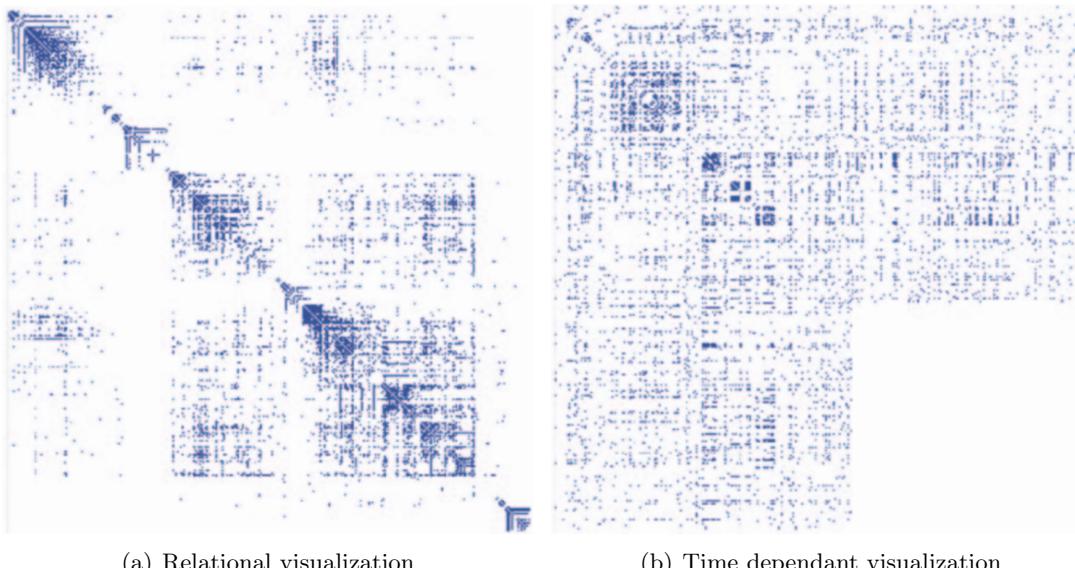


Figure 7: Matrix visualization of the facebook graph.

### 3.2 Dealing with missing values

Values can often miss from the data we have, because of various events (war, fire, ...). We must detect and correct these values with different method according with the domain we are working in.

Whatever the method used, it's good to keep track of these changes to know which are original data and which are modified ones.

- Set values to zero FIG 8(a)
- Interpolate based on existing data FIG 8(b)
- Omit missing data FIG 8(c)
- Interpolation with track kept 8(d)

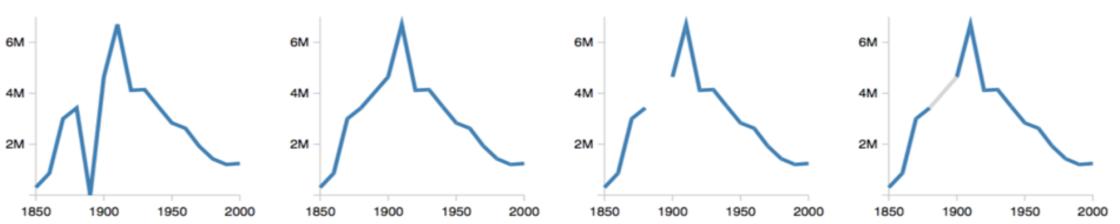


Figure 8: To deal with missing values.

### 3.3 General procedure

Once the data are well wrangled and before trying to analyse them we must take care of two more steps:

1. **Deal with uncertain data** (can arise from measurement errors, wrong sampling strategies, etc.)
2. **Parse/trasform data** (with aggregation and reduction techniques) to obtain meaningful records

It's always ideal to have the code and/or the documentation about the dataset you are analyzing (provenance).

## 4 Data Variety

The “3 Vs” of Big Data: *Volume*, *Velocity* and *Variety*. In this course, we don't address the *Volume* and *Velocity* parts (A course on Database does). Since there is a lot of variety in the data, we need to prepare the data. This flow is called **ETL**:

- **Extract** from the *source(s)*.
- **Transform** data at the source, sink, or in a *staging area*.
- **Load** data into the *sink*.

This variety of the data comes, in a first place, from the many different sources from which we extract them: *files*, *databases*, *logs*, ... Each of these sources uses (or not!) its proper convention and can contains structured (DB), semi-structured (logs) or unstructured (web page) data.

## 4.1 Role of Schema

The **Schema**, which specifies the *structure* and *types* of data repository, is changing. Traditional databases are **schema-on-write**, *i.e.* you cannot load data into a table without a schema. But new data stores (NoSQL for example) are **schema-on-read** or **schemaless**.

- **Schema-on-write** is typically SQL, where we must create a table before inserting data in our system. Data must scale the defined schema and this is both the strength and weakness of the system. Strength because the data is perfectly oriented and respect the constraints we establish. Weakness because schemas are always subjective in some ways and data (which are perfectly corrects) may not fit with it.
- **Schema-on-read** is for instance XML, where you create the schema according with the data you read.
- Youtube and Google Cache where the first **schemaless** data system. Without schema, everything is simply stored as a string and we need a parser to return a typed data.

## 4.2 Examples of data

### 4.2.1 XML and DOM

The XML data are used mostly with HTML and specifies the data structure. An XML schema can be applied to interpret the XML data and specifies the **data types**. Figure 9 shows the XML data 9(a) and the schema 9(b) used to parse and type the data.

```

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="unqualified">
<xsd:complexType name="location">
  <xsd:sequence>
    <xsd:element name="latitude" type="xsd:decimal"/>
    <xsd:element name="longitude" type="xsd:decimal"/>
  </xsd:sequence>
</xsd:complexType name="location">

```

(a) XML data

(b) XML schema

Figure 9: Example of XML.

The XML is a text format that encodes **DOM** (Document-Object Models). It's a data structure often used by Web pages. The DOM is tree-structured. An example of a DOM is given in Figure 10.

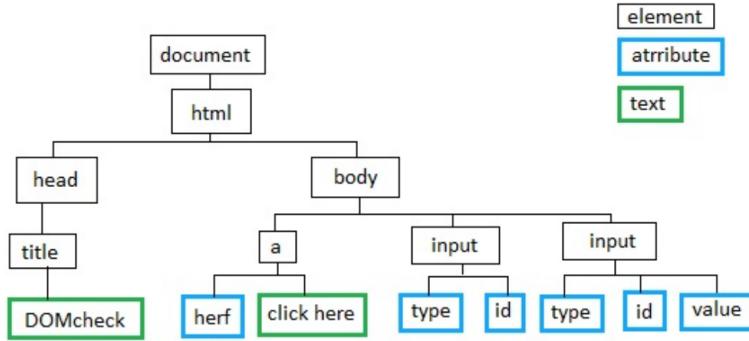


Figure 10: Example of a DOM tree for an HTML Web page.

The XML schema allows a database to interpret the data when running queries. It can do arithmetic or range queries on numerical values, for example.

#### 4.2.2 JSON

**JSON** stands for Javascript Object Notation. It's a schemaless data (schema support was added later). An example of JSON data is shown in Figure 11

```
{
  "firstName": "John",
  "lastName": "Smith",
  "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
}
```

Figure 11: Example of a JSON data.

JSON is typically used to represent **hierarchical data structures** directly in the target language (Javascript or Java at the beginning). The transformation on the data are procedural in the target languages. It is often easier for some tasks, but it can be painful for some of them: for example schema changes.

#### 4.2.3 Tabular data

A Tabular Data is simply data put into a table such as CSV or TSV. Definition of a table:

- A **table** is a collection of **rows** and **columns**.

- Each row has an **index**.
- Each column has a **name**.
- A **cell** is specified by an (index, name) pair.
- A cell may or may not have a **value**.

It's a very simple yet powerful data type. For example, the sensors usually output data in the form of time series, transformed into a tabular format. However, a system dealing with sensor data should:

- support both long-term (**trend**) and short-term (**real-time**) queries
- have **low latency** but also efficient. It should use **real-time indexing** for longer-term queries.
- support triggers (**alerts**) for a variety of conditions.

Therefore, the **complexity of a data format** does not determine the **complexity of the system required to properly handle it**.

#### 4.2.4 Log files

The log files are simple text files giving information about process. The daemons, such as `httpd`, `mysqld` or `syslogd`, usually create logs. `syslog` was developed by Eric Allman. It's a way for devices to send event messages to a server that will log all the events. Splunk is a company that built a successfull business model around the `syslog` events.

#### 4.2.5 Binary formats

They are often the key to performance because we **avoid expensive parsing**. The modern formats even support nested structures, various level of schema enforcement, **compression**, etc. Some examples: Protocol Buffers (Google), Avro, Parquet, etc.

### 4.3 Processing the data (JSON and XML)

In order to process XML, we can use the DOM. It can also be used to process JSON data. The DOM is very easy to work with: all the data are directly accessible by links. The problem is that we **might not care about most of the data** and if the data are big, they **might not fit into the RAM**. In order to deal with these two problems, we can use a **SAX** parser which is an event-driven parser. It will find all the **open-close-tag events** in an XML document and will do callbacks to user code.

- + User code can respond to only a subset of events corresponding to the tag it is interested in.
- + User code can correctly compute aggregates from the data rather than create a record for each tag.

- + User code can implement flexible error recovery strategies for ill-formed XML.
- User code must implement a state machine to keep track of “where it is” in the DOM tree.

For JSON, most parsers construct the “DOM” directly. But there are a few SAX-style parsers: Jackson, JSON-simple, etc. Sometimes **SAX-style is the way to handle ill-formed datasets**, an endless array of objects for example.

## 4.4 HTML and Web Services

### 4.4.1 HTML

Internet contains an “enormous” amount of data. Some crawlers such as Common Crawl dataset contains about 1.82 billions web pages (for 145 TB). We can use different tools to crawl data from the web. Examples for Python: BeautifulSoup, Requests, Scrapy, etc.

Most of the time, the Web pages are considered as unstructured data. But you can find some semi-structured data, *e.g.* Google WebTables. Some big “internet” companies (Google, Yahoo, Yandex and Microsoft) are sponsoring a project called **schema.org** to create structured or semi-structured Web pages. A core vocabulary for the type of fields is given. schema.org is more and more used. It’s also used by knowledge bases such as Google Knowledge Graph. **WikiData** is a community project to create an open database of structured data taken from Wikipedia.

### 4.4.2 Web Services

Screen-scraping the content of a large website was possible before, but become more and more difficult nowadays. This is mainly due to the content ”hidden” behind a form or an authentication. Take for example facebook without account, or the IS-academia course page if you do not select a semester. Therefore big companies are providing Web Service APIs<sup>1</sup> to retrieve data from their website. There are two kinds of Web Services:

- The old way: XML-based RPC-style messages: SOAP
- The new way: REST-style stateless interactions, URLs encode state

#### 4.4.2.1 RPC

The SOAP RPC<sup>2</sup> messages typically encode arguments that are presented to the calling program as parameters and return values. HTTP POST/GET are used to communicate.

---

<sup>1</sup>Application Program Interface: Set of subroutine definitions, protocols, and tools for building software and applications. In this particular case, the APIs are used to retrieve the data from the Web page, *e.g.* Facebook API to retrieve the contacts.

<sup>2</sup>SOAP = Simple Object Access Protocol, RPC = Remote Procedure Call

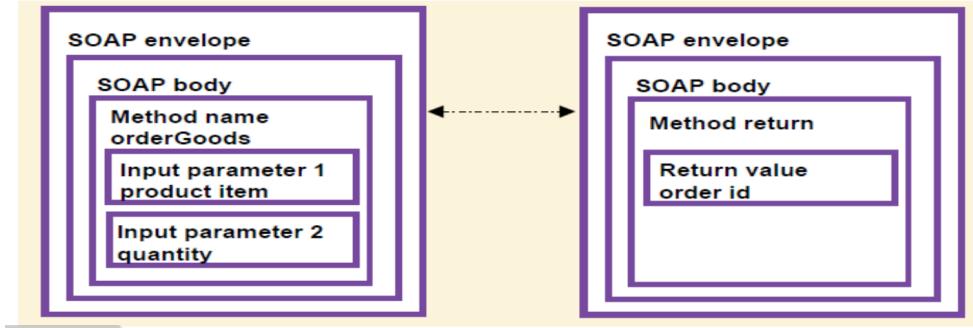


Figure 12: Example of a SOAP RPC exchange.

This kind of procedure (same for XML-RPC) requires a request-response cycle. This often leads to longer “conversations”. The RPC-style is being quickly superseded by newer and more user-friendly technologies.

In **RPC systems**, the design emphasis is on **verbs**. It uses functions such as `getUser()`, `addUser()`, etc.

#### 4.4.2.2 REST

REST<sup>3</sup> is a **stateless** client/server protocol. The principles are:

1. Each message in the protocol contains all the information needed by the receiver to understand and/or process it. This constraint attempts to “*keep things simple*” and avoids needless complexity.
2. Set of Uniquely Addressable Resources
  - “*Everything is a Resource*” in a RESTful system
  - Requires universal syntax for resource identification, *e.g.* URI.
3. Set of Well-Defined Operations that can be applied to all resources
  - In the context of HTTP (REST APIs), the primary methods are: **POST**, **GET**, **PUT**, and **DELETE**  
These are similar (but not exactly) to the database notion of CRUD (Create, Read, Update, and Delete)
4. The use of Hypermedia both for Application Information and State Transitions
  - Resources are typically stored in a structured data format that supports hypermedia links, such as XHTML or JSON.

In **REST systems**, the design emphasis is on **nouns**. It uses the HTTP Protocols (POST, GET, PUT, and DELETE) a *User*, a *Location*, etc.

---

<sup>3</sup>REpresentation State Transfer

## 5 Statistics on the Data

When we explore and analyze data, it would be great if we only had to look at some statistics numbers and make automatically a conclusion about them. Sadly, it's not the case. At all.

### 5.1 Example of famous mistakes due to statistics

#### 5.1.1 Anscombe's quartet: Sensivity of outliers & Robust statistics

The FIG 13 show four different data distribution that present, despite all of that, the same means on  $x$  and  $y$ , the same variance on  $x$  and  $y$ , and, thus, the same linear regression function. This is due to the statistics used to define them.

- Min, Max, Mean, Standard Deviation (Std) and Range are sensitives to outliers and then are **not robust statistics**.
- Median, quartiles, (and others) are not sensitives and then are said to be **robust statistics**.

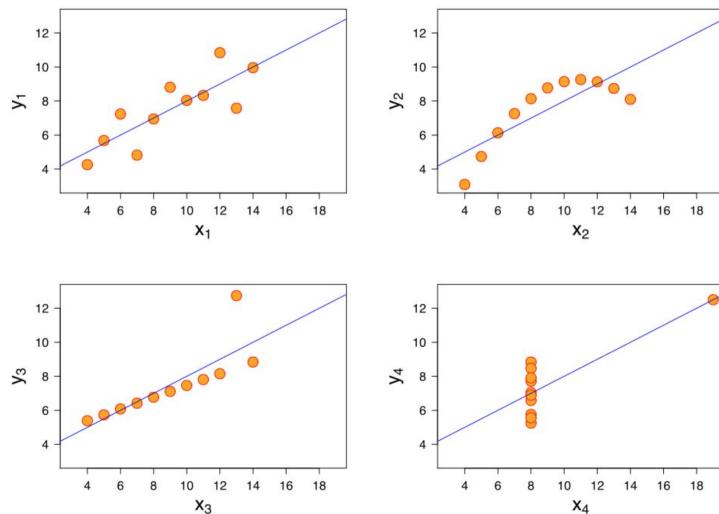


Figure 13: Anscombe's quartet

#### 5.1.2 Simpson's paradox: aggregation of data

Certain tendencies can appear, disappear or even reverse themself when aggregating the data! This was the case when media started blaming Berkley of being unfair with women applications (looking at left table of FIG 14). After further investigation (and de-aggregation of the data) it appeared that at the opposite... Berklay was unfair with men! (Right table of the same FIG).

This paradox comes from the fact that women (according with these tables) tended to apply for more competitive departments, with lower rates of admission. **When aggregating the data, we lost this subtlety and then draw a wrong conclusion.**

Simpson's paradox can appear in a lot of cases and can be very hard to detect. The [wikipedia page of Simpson's paradox](#) describes a lot of examples and, for the ones interested, a great book relates lots of statistical errors that drove to miscarriages of justice: [Leila Schneps and Coralie Colmez, Math on Trial: How Numbers Get Used and Abused in the Courtroom](#)

| Department | Men        |          | Women      |          |
|------------|------------|----------|------------|----------|
|            | Applicants | Admitted | Applicants | Admitted |
| <b>A</b>   | 825        | 62%      | 108        | 82%      |
| <b>B</b>   | 560        | 63%      | 25         | 68%      |
| <b>C</b>   | 325        | 37%      | 593        | 34%      |
| <b>D</b>   | 417        | 33%      | 375        | 35%      |
| <b>E</b>   | 191        | 28%      | 393        | 24%      |
| <b>F</b>   | 373        | 6%       | 341        | 7%       |

Figure 14: Berkley admission tables of 1973

## 5.2 Refresh of basic stats

Definitions:

- **Probabilities:** mathematical theory that describes uncertainty.
- **Statistics:** Set of techniques for extracting useful info from data

**Probability and Statistics** are related areas of mathematics which concern themselves with analyzing the relative frequency of events. Still, there are fundamental differences in the way they see the world:

Probability deals with predicting the likelihood of future events, while statistics involves the analysis of the frequency of past events.

Probability is primarily a theoretical branch of mathematics, which studies the consequences of mathematical definitions. Statistics is primarily an applied branch of mathematics, which tries to make sense of observations in the real world.

Steven S. Skiena, "Calculated Bets", Cambridge University Press, 2001

### 5.2.1 Bayes Theorem

The theorem express the very intuitiv statement that:

The probability of observing event A and B is the probability of observing B multiplied by the probability of observing A knowing that B occurred.

Mathematicaly it's expressed as:

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A) \quad (1)$$

Or equivalently

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (2)$$

More about Bayes Theorem on [wikipedia](#).

### 5.2.2 Random Variables

A **random variable** is a quantity that can take various values, each one associated with a probability of apparition. The sum of these probabilities will always be 1.

$$X : \Omega \rightarrow E \quad (3)$$

$\Omega$  being a probability space and  $E$  a measurable space (usually  $E = \mathbb{R}$ ).

Any random variable can be described by its [cumulative distribution function](#), which describes the probability that the random variable will be less than or equal to a certain value.

Two **independent variables** are defined as

$$\mathbb{P}(A \cap B) = \mathbb{P}(A) \cdot \mathbb{P}(B). \quad (4)$$

or equivalently (by Bayes Theorem)

$$\mathbb{P}(A | B) = \mathbb{P}(A) \quad (5)$$

More about Bayes Theorem on [wikipedia](#).

### 5.2.3 Law of Large Numbers

The Law of Large Numbers links, in some way, the probability to the statistics. It's, again, a very intuitive statement, even if not so easy to prove (as always in mathematics).

In probability theory, the **law of large numbers** (LLN) is a theorem that describes the result of performing the same experiment a large number of times. According to the law, **the average of the results obtained from a large number of trials should be close to the expected value**, and will tend to become closer as more trials are performed.

[Wikipedia](#)

A common mistake is to deduce that, in the case of playing heads or tails, observing a lot of time **heads** increase the probability of observing **tails**. This is absolutely wrong. The variables are perfectly independent and, according with Eq 5, the probability stays exactly 50%.

### 5.3 Measurement on Samples

Measurement on Samples:

Many datasets are samples from an infinite population. We have access only to a **sample** of a whole population. Therefore, measurements are hard because:

- Samples measurements have **variance**: Variation between samples
- Samples measurements have **bias**: Systematic variation from the population value.  
(Thermometer say temp is 10 degrees more than pop)

Normal Distribution, Mean and Variance:

Mean is the average of all values. Variance measure the width of a distribution. (EQUATION) Standard deviation = sqrt of variance. (IMAGE?)

Central Limit Theorem:

Distribution of the sum (or mean) of a set of  $n$  identically-distributed random variables  $X_i$  **approaches a normal distribution as  $n \rightarrow \infty$** .

Common parametric statistical test, like t-test and ANOVA, assume normally-distributed data, but **depend on sample mean and variance measures of the data**.

They typically work reasonably well for data that are not normally distributed as long as the samples are not too small!

Often the data are not normally distributed. (But the stat tools assume that.) Therefore, check the data!

Distributions:

**Poisson** Distrib of counts that occur at a certain rate.

- Observed frequency of a given term in corpus.
- Number of visits to a web site in a fixed time interval.
- Number of website clicks in an hour.

**Exponential** Interval between two such events.

**Zipf/Pareto/Yule distrib** govern the frequencies of different terms in a document, or web site visits.

**Binomial/Multinomial** govern the number of counts of event (e.g. die tosses = 6) out of  $n$  trials.

**You should understand the distribution of your data before applying a model!**

Rhine Paradox:

Joseph Rhine was a parapsychologist. Experiment: 10 hidden cards are red or blue. Result: 1 personne out of 1000 could guess the color of all 10 cards. Proba to guess everything right: 1/1024. WRONG! Stats can be useful.

Hypothesis Testing:

- We want to prove a hypothesis  $H_A$  by means of **disproving a null hypothesis  $H_0$** .
- A **test statistic** is some measurement we can make on the data which is likely to be **big under  $H_A$**  but **small under  $H_0$** .

## **Je pense que l'exemple pourrait etre ajouter.. Slides 18 - 21**

One-tailed takes care of only one tail of the distribution. The two-tailed takes care of both tails of the distribution. Add the image of the two-tailed test!

Testing is a bit like **finding the right recipe** based on these ingredients:

1. Question
2. Data type
3. Sample size
4. Variance known?
5. Variance of several groups equals?
6. ...

**Plenty of tables available!** PICTURE Slide 25!!

Multiple Hypothesis Testing:

If you perform experiments over and over, you're bound to find something. Significance level must be adjusted down when performing multiple hypothesis tests.

Familywise Error Rate:

Error rate when too many experiments are done.. Je te laisse developer =P. There are some corrections: Bonferroni and Sidak.

Non-Parametric Tests:

All the tests so far assume that the data are **normally distributed** and that the samples are **independent of each other and all have the same distribution.** (IID) They may be inaccurate if those assumptions are not met. Therefore, make sure the data satisfies the assumptions of the test we're using. Watch out for:

**Outliers** will corrupt many tests that use variance estimates.

**Correlated values as samples** , e.g. if you repeated measurements on the same subjective

**Skewed (bias) distributions** give invalid results.

K-S test:

K-S (Kolmogorov-Smirnov) test is a very useful test for checking whether two (continuous or discrete) distributions are the same.

- In the **one-sided test**, an observed distribution (e.g. some observed values or a histogram) is compared against a reference distribution (e.g., power-law).
- In the **two-sided test**, two observed distributions are compared.
- The K-S statistic is just the **max distance between the CDFs** (Cumulative Distribution Function) of the two distributions.
- The K-S test can be used to test **whether a data sample has anormal distribution** or not.
- Thus it can be used as a sanity check for any common parametric test (which assumes normally-distributed data).
- It can also be used to compare distributions of data values in large data pipeline:  
**Most errors will distort the distribution of a data parameter and a K-S test can detect this.**

This test is expensive!

**WE NEED TO ADD MORE STUFF ON THE K-S TEST. CHECK ON WIKIPEDIA.. =)**