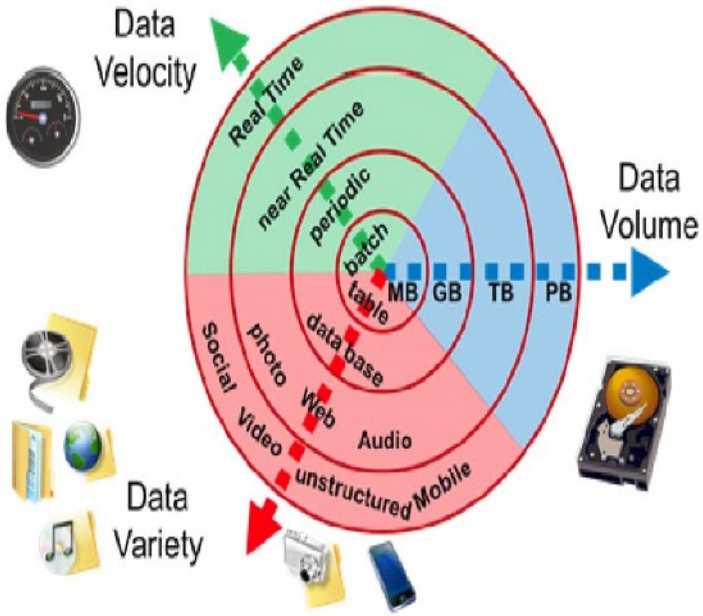
# DATA SCIENCE MEI/1

**University of Beira Interior**
Department of Informatics

Hugo Pedro Proença,
hugomcp@di.ubi.pt, **2021/22**

# Data Science File Formats

- **Unified analytics engines** (such as *Apache Sparks*) are particularly suited for large-scale data processing.
  - They provide high-level APIs in different languages, such as Java, Scala, **Python** and R
  - Optimized engines that support general execution graphs.
- Often, they also support a rich set of higher-level tools for structured data processing, machine learning, graph processing, incremental computation and stream processing.
- These frameworks also support many different data formats, such as the ubiquitous CSV format and the friendly web format JSON.




Source: https://en.wikipedia.org/wiki/Big_data#/media/File:Big_Data.png

# Data Science File Formats

- Some of the most popular formats are "**CSV**", "**JSON**", "**Parquet**" and "**Avro**".
- However, depending of the specific application, there are many other formats that might have to be considered:
  - Containers: TAR, GZIP, ZIP
  - Databases: XML
  - Geospatial vector data: SHP, GeoJSON, KML, DBF, NetCDF
  - Geospatial raster data: GeoTIFF/TIFF, NetCDF, HDF-EOS
  - Videos: MOV, MPEG, AVI, MXF
  - Sounds: WAVE, AIFF, MP3, MXF
  - Statistics: ASCII, DTA, POR, SAS, SAV
  - Images: TIFF, JPEG 2000, PDF, PNG, GIF, BMP
  - Text: XML, PDF/A, HTML, ASCII, UTF-8
  - Web archive: WARC

# Data Science File Formats

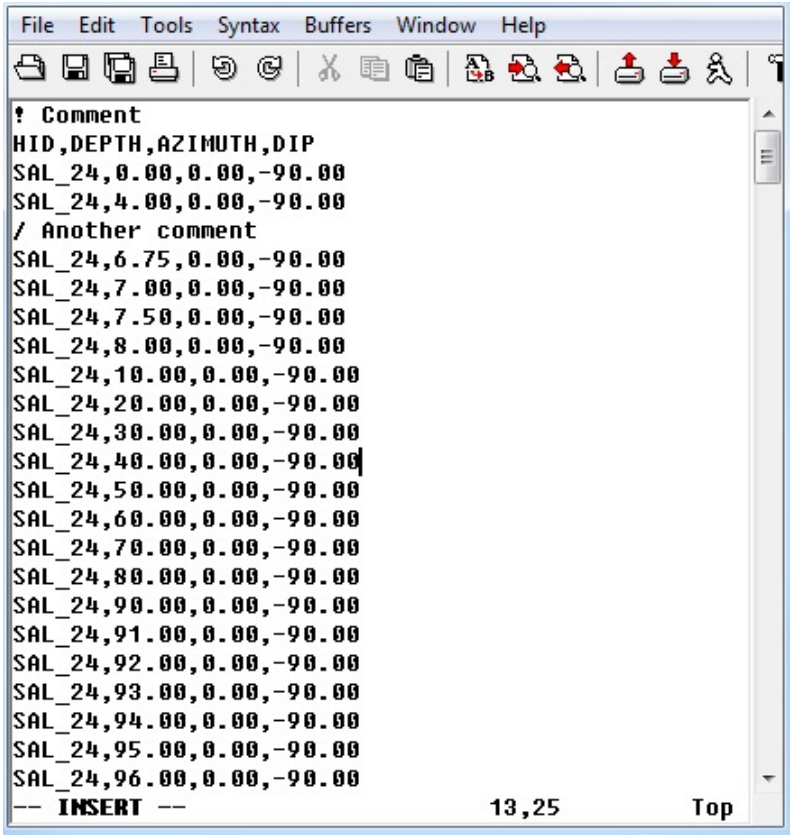
## ☐ Tabular Data

- ☐ Data provided in bidimensional structures (tables or spreadsheets), is by far the most common format for presenting, analyzing, and storing data. However, the most common spreadsheet file formats (e.g., *Microsoft Excel*) are far from ideal for sharing, preserving, and reusing data
- ☐ Such formats are not easily machine readable and may be difficult or impossible to read if the specific software tools used to create them are significantly upgraded or outmoded.
- ☐ Using delimited text formats is a much better way to ensure that tabular data are readable in the future.
- ☐ A delimited text file is an ASCII-encoded file used to store data in which each line is uniquely represented and has fields separated by a special character—the **delimiter**.
- ☐ Common delimiters are the comma, tab, and colon. In a file with comma-separated values (a CSV file), the data values are separated using commas as a delimiter. One benefit of being incredibly common and simple to use is that most database and spreadsheet programs are able to read in or export data in a text-delimited format.

# Data Science File Formats: CSV

## Comma-separated Value Files (CSV)

- CSV files are usually used to exchange tabular data between systems using plain text.
- It is a row-based file format, which means that each row of the file is a row in the table.
- A CSV file typically contains a header row that provides the column names for the data
  - When this does not happens, data are considered only partially structured.
- CSV files may not initially contain hierarchical or relational data.
- Data connections are usually established using multiple CSV files.
- Foreign keys are stored in columns of one or more files, but connections between these files are not expressed by the format itself.



# CSV Format

- There are different advantages/disadvantages of using this format:
  - **Advantages:**
    - CSV is human-readable and easy to edit manually;
    - CSV is compact. For example, in XML format, we have to start a tag and end a tag for each column in each row;
      - In opposition, in CSV format , the column headers are written only once; CSV is human-readable and easy to edit manually;
    - CSV provides a simple scheme;
    - CSV can be processed by almost all existing applications;
    - CSV is easy to implement and parse;

# CSV Format

- There are different advantages/disadvantages of using this format:
  - **Disadvantages:**
    - CSV provides no support for column types. No difference between text and numeric columns;
    - There is no standard way to present binary data;
    - There are often problems with CSV import (for example, no difference between NULL and quotes);
    - It provides poor support for special characters;
    - Lack of a universal standard.
- Despite these limitations, CSV format is a highly popular choice for data exchange as they are supported by a wide range of business, consumer, and scientific applications.
- Most batch and streaming frameworks (e.g. Spark and MR) initially support serialization and deserialization of CSV files and offer ways to add a schema while reading.

# Data Science File Formats: JSON

- **JSON** is an acronym for "**JavaScript Object Notation**"
- In this format, data are presented as key-value pairs in a partially structured format.
- JSON format is often compared to XML because it can store data in a hierarchical format.
- Both formats are user-readable, but JSON documents are typically much smaller than XML.
- JSON format is commonly used in network communication, especially with the rise of REST-based web services.


```
{
    name: "DCIM",
    type: "folder",
    path: "DCIM",
  - items: [
      - {
            name: "Snaphots",
            type: "folder",
            path: "DCIM/Snaphots",
            items: [ ]
        },
      - {
            name: "SoundRecorder",
            type: "folder",
            path: "DCIM/SoundRecorder",
            items: [ ]
        },
      - {
            name: "VideoRecorder",
            type: "folder",
            path: "DCIM/VideoRecorder",
          - items: [
              - {
                    name: "2018-03-06_13.36.mov",
                    type: "file",
                    path: "DCIM/VideoRecorder/2018-03-06_13.36.mov",
                    size: 40351122
```

# JSON Format

- Since much data is already transmitted in JSON format, most web languages initially support JSON. With such kind huge support, JSON is used to represent data structures, exchange formats for many data warehouses.
- Many streaming packages and modules support JSON serialization and deserialization.
- While the data contained in JSON documents can ultimately be stored in more performance-optimized formats such as Parquet or Avro, they serve as raw data, which is a very important feature for data processing (if necessary).

# JSON Format

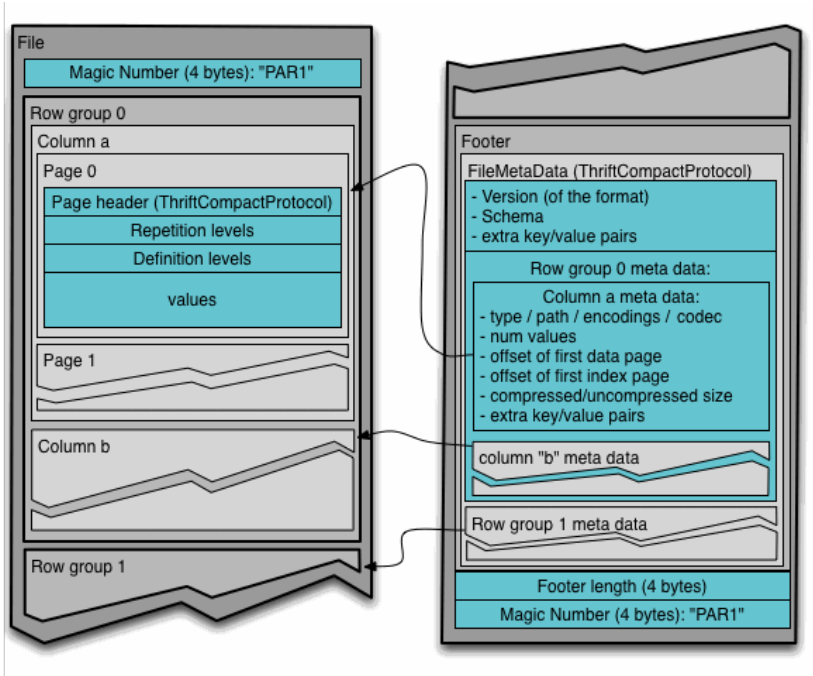
- There are different advantages/disadvantages of using this format:
  - **Advantages:**
    - JSON supports hierarchical structures, simplifying the storage of related data in a single document and presenting complex relationships;
    - Most languages provide simplified JSON serialization libraries or built-in support for JSON serialization/deserialization;
    - JSON supports lists of objects, which avoids chaotic list conversion into relational data models;
    - JSON is a widely used file format for NoSQL databases such as MongoDB, Couchbase and Azure Cosmos DB;
    - There is built-in support in most modern Data Science tools;

# JSON Format

- There are different advantages/disadvantages of using this format:
  - **Disadvantages:**
    - JSON is known to consumes substantiual computational resources (memory) due to repeatable column names;
    - There is poor support for special characters;
    - JSON files are not very splittable;
    - JSON format lacks indexing;
    - It is less compact as compared to over binary formats.
- In opposition to CSV, this format can store any of the following data types: (strings, numbers, objects, arrays and Booleans).
- In summary, it is most commonly used for transmitting data in web applications (e.g., sending some data from the server to the client, so it can be displayed on a web page, or vice versa).

# Data Science File Formats: Parquet

- **Apache Parquet** is a file format designed to support fast data processing for complex data, with several notable characteristics.
- In this format, as the data are stored in columns, they can be highly compressed (compression algorithms work better with data with low entropy of information, which is usually contained in columns) and can be separated.
- Its developers of the format claim that this storage format is ideal for solving Big Data problems.




Source: https://parquet.apache.org/documentation/latest/

# Parquet Format

- Unlike CSV and JSON, **Parquet** files are binary files that also contain metadata about their contents.
  - Therefore, without reading/parsing the contents of the file(s), Spark can simply rely on metadata to determine column names, compression/encoding, data types, and even some basic statistical characteristics.
  - Column metadata for a Parquet file is stored at the end of the file, which allows for fast, single-pass writing.
- Parquet is optimized for the paradigm Write Once Read Many (WORM).
  - As a result, it writes slowly but reads incredibly quickly, especially when we only access a subset of columns.
  - It is a popular choice for heavy workloads when reading portions of data.
  - In opposition, in cases where we need to work with the whole rows of data, other formats (e.g., CSV or AVRO) should be preferred.

# Parquet Format

- There are different advantages/disadvantages of using this format:
  - **Advantages:**
    - Being a columnar format, only the required columns are retrieved/read, which reduces disk I/O. The concept is called projection pushdown;
    - The scheme travels with the data, so the data is self-describing;
    - Although it is designed for HDFS, data can be stored on other file systems such as GlusterFs or NFS;
    - There is built-in support in Spark makes it easy to simply take and save a file in storage;
    - Parquet provides very good compression up to 75% when using even compression formats like snappy;
    - The practice shows that this format is the fastest for read-heavy processes compared to other file formats;
    - It is particularly well suited for data storage solutions where aggregation on a particular column over a huge set of data is required;

# Parquet Format

- There are different advantages/disadvantages of using this format:
  - **Disadvantages:**
    - The column-based design makes you think about the schema and data types. In practice, it reduces independency between the data source creation and use;
    - Parquet does not always have built-in support in tools other than Spark;
    - However, the main problem is that it does not support data modification (Parquet files are immutable) and scheme evolution.
    - Of course, there are tools that know how to combine different schemas in case of changes over time
    - However, we can only change something in an existing file by overwriting it.

# Data Science File Formats: Avro

- This format was released by the Hadoop working group in 2009.
- It is a row-based format that has a high degree of splitting.
- It is often described as a data serialization system similar to Java Serialization. The schema is stored in JSON format, while the data is stored in binary format, which minimizes file size and maximizes efficiency.
- Avro has reliable support for schema evolution by managing added, missing, and changed fields. This allows old software to read new data, and new software to read old data — it is a critical feature if your data can change.

**Avro JSON schema sample**

```
{
  "type": "record",
  "name": "Event",
  "namespace": "com.liveperson.example",
  "doc": "Example event",
  "fields":[{ "name": "version", "type": "string",  "default": "1" },
            { "name": "id", "type": "string", "default": "Unknown"},
            {"name": "time","type": "long","default": -1},
            {"name": "body","type": "string","default": "no body"},
            {"name": "color","type":
                    { "type": "enum", "name": "Color",
                      "symbols": ["NO_COLOR", "BLUE", "BLACK", "WHITE", "PINK"]},
                      "default": "NO_COLOR" }
    ]
}
```

# Avro Format

- The possibility to manage scheme evolution allows components to be updated independently, at different times, with a low risk of incompatibility.
  - This eliminates the need for applications to write if-else statements to handle different versions of schema and eliminates the need for the developer to look at old code to understand the old schema.
  - Since all versions of the schema are stored in a human-readable JSON header, it is easy to understand all the fields available to you.
- Since the schema is stored in JSON and the data is stored in binary form, Avro is a relatively compact option for both permanent storage and wire transfer.
- Also, as it is a row-based format, it is the preferred format for handling large amounts of records as it is easy to add new rows.

# Avro Format

- There are different advantages/disadvantages of using this format:
  - **Advantages:**
    - Avro is a linguistic-neutral serialization of data.
    - Avro stores the schema in a file header, so the data is self-describing;
    - It is easy and fast to perform data serialization and deserialization, which can provide very good ingestion performance;
    - Avro files also contain synchronization markers to separate blocks. This makes it highly splittable;
    - Files formatted in Avro are splittable and compressible and are therefore a good candidate for data storage in the Hadoop ecosystem;
    - The schema used to read Avro files does not necessarily have to be the same as the one used to write the files. This allows new fields to be added independently of each other;

# Avro Format

- There are different advantages/disadvantages of using this format:
  - **Disadvantages:**
    - Perhaps the main disdavantage, is the rexduced Independence between the data schema creation process and the data utilization. Everytime we need to access to data in this format, we have to think about the schema and data types;
    - Being stored in binary format, its data is not human-readable;
    - Being relativelly recente, it is not yet integrated into most programming languages.