Module 3. Data exploration, transformation, and cleaning for later use

BARÇA INNOVATION HUB
Universitas

- Alternative Extract, Transform and Load (ETL)
- References

Alternative Extract, Transform and Load (ETL)

In data analytics, more specifically in areas of analytics such as data storage or the creation of automated data management and movement processes, there is a process for the correct integration and transformation of a specific database. This process is commonly known as ETL, an acronym for extract, transform and load.

In general terms, this process consists of extracting information from the different data sources and suppliers; transforming what is needed so that they fit the format and requirements of the database to which we want to send that information, and finally, automating the load of information to the database so that it can be used by professionals who require that information. These processes are not usually par of the responsibilities of a sports scientist as they are very specific tasks and require training and experience in a field outside the range of skills of a sports scientist. However, in our professional work we also have to perform processes that allow us to structure our data in the best possible way so that it can be used most efficiently.

For example, within the club where you work, the team coach or a member of the board wants to know how the team players are evolving throughout the season, as they know that tests are being carried out to assess their physical condition. They make this request once about every two weeks. You could go to each of the tools you used to record the results, download them and analyse them with tools such as Excel or RStuido itself (Allaire, 2021), but as we have mentioned, if this request is repeated frequently, you would better organize and store the data systematically so that the process of providing information is as efficient as possible.

This does not require using complex databases if you do not have that storage capacity. An Excel workbook can be an extremely valuable resource to have the information available and in a format that allows for its subsequent use with guarantees.

In order to achieve this goal, we will teach you how to use a process with the same acronym as the ETL described above, but specific to the needs and requirements of a sports scientist.

- Data exploration: part of the exploration is also present the descriptive analysis that we have mentioned in the previous module, but in this case we refer to the format of our data (how the data table is organized, what type of variables there are, how to detect errors, etc.).

- Data transformation: based on the previous step, we need to change the data format or table settings to make it as efficient as possible.

- Data cleaning: this involves correcting the errors that we have detected so that the subsequent analyses are correct.

In this module we will look into the most common options for each of the steps in this process, but you should bear in mind that the range of possibilities is very extensive and is related to the type of data to work with (time series, data, summary, etc.). Watching the video material offered during the modules is essential to get familiar with the possible options offered by RStudio (Allaire, 2021) in this regard.

Self-assessment:

The process commonly known as ETL consists of:

☐ extracting the information from the different data sources and suppliers

- [ ] making the corresponding transformations so that they fit the format and requirements of the database to which we want to send that information
- [ ] automating the load of information to the database so that it can be used by professionals who require that information
- [ ] presenting the information to our suppliers who need it for further analysis

SUBMIT

Data organization

Before we dive into each of the steps mentioned above, we want to emphasize the importance of data logging. In this case, we will not focus so much on the consistency and selection of tools that allow us to trust data quality. Instead, we will focus on how to store that data, how it should be structured, and how consistent the record format should be. The features we will highlight below are easily applicable to basic data-log files such as Excel, but the same principles can be applied to more complex storage systems.

The same type of information can be found in different formats. For example, you could have an Excel workbook where the data is stored in the first sheet in "width" format (image A, figure 1) in which each of the variables has its corresponding column, or in "long" format (image B, figure 1) in which there is a column with an identifier of the questionnaire variable and another column with the values of each of the answers.

Figure 1: Basic data records

A

Jugador	Fecha	Sueño	Fatiga	Dolor
A	1/1/24	3	5	3
B	1/1/24	4	2	2
C	1/1/24	3	5	1
D	1/1/24	5	4	1

B

Jugador	Fecha	Variable	Valor
A	1/1/24	Sueño	3
B	1/1/24	Sueño	4
C	1/1/24	Sueño	3
D	1/1/24	Sueño	5
A	1/1/24	Fatiga	5
B	1/1/24	Fatiga	2
C	1/1/24	Fatiga	5
D	1/1/24	Fatiga	4
A	1/1/24	Dolor	3
B	1/1/24	Dolor	2
C	1/1/24	Dolor	1
D	1/1/24	Dolor	1

Source: author's own adaptation on the basis of Grolemund and Wickham, 2017.

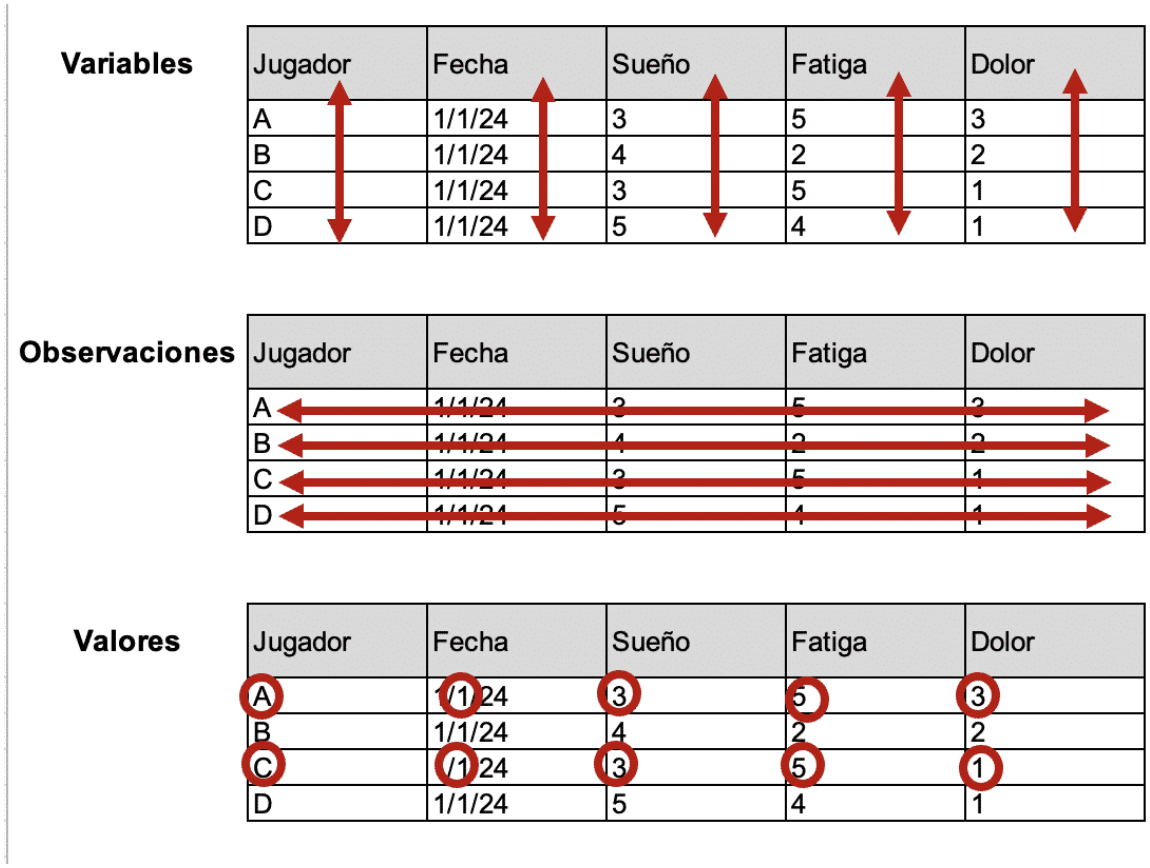
Depending on the type of analysis or visualization you need to perform, one format or another will be more convenient, but there are certain rules to make up a well-organized database (Grolemund and Wickham, 2017).

- Each variable must have its own column.

- Each observation must have its own row.
- Each value must have its own cell.

Based on the image above, you can see what each of these rules refers to with the following image (figure 2).

Figure 2: Rules for a well-organized database



Source: author's own adaptation on the basis of Grolemund and Wickham, 2017.

ONCE THESE FIRST RULES ARE MET, THERE ARE A NUMBER OF CONSIDERATIONS TO EXAMINE (BROMAN AND WOO, 2018).

- Be consistent: collect information in the same way every day. In the questionnaire mentioned above, for example, if you have a column with the variable "Name" or "Player", you should verify that the name you register is always the same. You should not use abbreviations or change the order of first or last name, etc.
    - Data crossing: This functionality will be covered later on in other courses, but being consistent is essential for efficient operation if you are looking to use information from different data sources. It is best practice to use fixed identifiers for each of the players or athletes and that these are the same in any of the data-logging systems you use.
    - Dates: this is usually a problematic data in terms of format consistency, since different data-recording software uses a variety of formats. Being consistent in this will go a long way in helping you avoid costly data transformation and cleansing.

- Choose the right nomenclature: you should avoid special characters and, if possible, spaces and capital letters. This should apply either to the variable/column name or the cell where values are inserted. For example, if you have recorded anthropometric values and you have a variable regarding the percentage of muscle mass, instead of calling your variable "% Muscle Mass", since this contains special characters and spaces, you could opt for "pct_muscle_mass".
    - If the nomenclatures becomes unintuitive, you can create documents that work as dictionaries of such variables.

- Avoid empty cells: In our field we are used to finding unrecorded data. For example, in an entire observation of a specific player sometimes it

happens that data is missing from all variables, while there could be a single variable or a full day for the whole team. You need to decide how to log that unrecorded data so that you can correctly identify it in future analyses.

    - You should also avoid file types such as those detailed in the first module, in which at the top of the Excel sheet there are X rows with information and then there is a data table. This type of format is not ideal for analysis.

- Do not perform calculations on the main file: you must keep the file where you record your data as clean as possible, that is, unprocessed. For any of the calculations you want to make, it will be a better option to use RStudio (Allaire, 2021). In this way you will also avoid losing information.

If you follow these tips, you will have a good structure to develop the next steps.

Self-assessment:

That each variable must have its own column, each observation must have its own row and each value must have its own cell are rules to consider a:

Type your answer here

SUBMIT

Data exploration

The first data exploration to do is in order to identify its format, whether you are dealing with a data table and each of the variables it contains or whether you will work with "objects", vectors or lists.

- “class()" function

To find out the type of data you will deal with, you can use the "class()" function. This function allows you to identify the class of the data and its format, so that you can use it appropriately later. Please note that functions need "arguments", i.e. an object to which the function can be applied, and also a series of arguments that each of the functions requires. In the examples below, you can see the arguments used in some simple functions.

For example, variables/columns could have different types of formats: they could be numerical, "character", categorical/factor, dates, etc. Depending on the type of variable, some calculations or others can be made, in the same way that each of the functions used will be specific to the data format to be processed.

Figure 3: Function applied to different objects

```r
class(data_rpe)
```

```
[1] "tbl_df"     "tbl"        "data.frame"
```

```r
class(data_rpe$RPE)
```

```
[1] "numeric"
```

```r
fecha_informe <- ymd("2024-01-01")
class(fecha_informe)
```

```
[1] "Date"
```

```r
idjugadores_lesionados <- c("12556","77857")
class(idjugadores_lesionados)
```

```
[1] "character"
```

Source: Author's own production

In the image above, you can see how the same function is applied to different objects within the "R" environment. First, it is applied to the object “data_rpe”. The result is displayed below the written code and indicates that it is a "data_frame", one of the most common "R" data table formats. The function is then applied again, but in this case to one of the variables in the "RPE" table that contains the players' responses in the default scale. Here you can see that this is a column that contains numerical values.

The last two cases illustrate how two objects are created. In the former, a single value is specified and the "ymd()" function is applied to it, which, generally speaking, formats the specific text entered as a date. In the latter, the "class()" function is used to verify that it is indeed the date format, so the conversion has been correct. You could use this object for your daily reports, specifying the date on which you want to make the report. You could also use this object to filter for the data in subsequent steps.

Finally, the second object created is a vector, as it contains more than one value and its values are in "character" format. Even if they are numbers, when written in quotation marks they get "character” format. According to the example above, if you do not want injured players to appear in your report, you could use this vector to filter out the unwanted players from your data source. For this step to work properly, the format of the database column containing the player

identifiers must also be "character". Otherwise, the filter would not work properly. Hence the usefulness of knowing the data format.

- “str()” function

Figure 4: “*str()*” function

```
```{r}
str(data_rpe)
```
```

```
tibble [3,178 × 4] (S3: tbl_df/tbl/data.frame)
 $ Name_id: int [1:3178] 1587 1792 95 445 1035 2622 1282 1807 1565
 $ Date   : Date[1:3178], format: "2020-08-11" ...
 $ HORA   : POSIXct[1:3178], format: "1899-12-31 20:19:55" ...
 $ RPE    : num [1:3178] 5.5 4 2 6 2 3 6 4 4.5 4 ...
```

Source: Author's own production

In this code extract and its result you can see the usefulness of the "str()" function that represents the word structure, that is, it seeks to know how the database is organized. It will give information about whether it is a table or another type of object, the number of rows and columns (3178 rows and 4 columns) and the characteristics of each variable. In addition, you will see the first observations of each variable which give an idea of the data that will come up.

- “ncol()" function: this function allows you to know the number of columns in the data table, in case you do not need to know all the information provided by the "str()" function.

- "nrow()" function: it has the same functionality as the previous one, but in this case you would get the number of rows.

- "dim()" function: it is a combination of the previous two; it will show you the number of rows and columns.

- “colnames()" function

  This function serves to know the name of the columns in the data table if you use it only with the argument of the object for which you want to know the names of the columns. This same function can be used to rename columns, but this corresponds to the data transformation section.

- “*summary()*” function

Figure 5: “*summary()*” function

```
summary(data_rpe)
```

```
    Name_id           Date
Min.   :   1.0   Min.   :2020-08-11
1st Qu.: 795.2   1st Qu.:2020-09-11
Median :1589.5   Median :2020-12-02
Mean   :1589.5   Mean   :2020-12-05
3rd Qu.:2383.8   3rd Qu.:2021-02-19
Max.   :3178.0   Max.   :2021-05-19

      RPE
Min.   : 0.000
1st Qu.: 4.000
Median : 5.000
Mean   : 5.191
3rd Qu.: 6.500
Max.   :10.000
```

Source: Author's own production

As you can see in the image, this function allows you to have a numerical summary of the variables in the data table. This information may be useful for some of the variables, such as the "RPE", but irrelevant in the case of the "Name_id" column, since although they are numbers, they are identifiers, so descriptive

parameters such as the mean or maximum have no informative value.

- "head()" function

  It displays the first 6 rows of the data table and allows you to get a first view of the data you are dealing with.

- “unique(data_rpe$Date)" function

  For one of the columns in the data table or an object such as a vector, this function allows you to know the number of unique values in that variable. For example, in the case of a table with "RPE" information, there will be a multitude of observations with the same date, since each player who answers the questionnaire on the same day will have the same date associated with it. If you want to see on which (unique) dates the questionnaire was answered, you must use the "unique()" function.

- “view()" function

  This function is very useful when you are new to RStudio, as it allows you to view the table or object in

a format much closer to what you may be used to, in a separate window in the RStudio interface (Allaire, 2021).

Data transformation

This point in the data analysis process can be as complex as the project you are working on; however, knowing the basic functionalities that allow you to modify the structure and content of your data source will allow you to achieve great results from the first projects you conduct. A list and brief description of the most common functions of data transformation will be presented below. The video content of this module will be instrumental to understand the applicability of these features in the context of the sport scientist and will allow you to move forward in the process.

As you may have seen in the videos, depending on the type of function you want to use, you will have to install one package or another.

- Columns transformation
  - "colnames()": mentioned above, this function will allow you to rename the columns for later use.
  - "as.factor()"/"as.numeric()": these allow you to change the format of the column values to the desired format.
  - “arrange()": this function allows you to sort the columns according to the parameters you want.

- Variable reduction or selection and observations
  - “select()": allows you to select the number of columns you want.

- “filter()": in this case, observations that meet the requirements you are looking for will be filtered.

- Creation of new columns:
  - "mutate()": this is one of the most commonly used features when starting to work with a database to create new columns or derivatives of existing ones.

- Change in the organization of the data within the table
  - "pivot_longer()": allows you to carry out transformations of the organization of the data such as the one you have seen in the first image of this module, going from a "wide" format to a "long" format, where there are variables with identifiers and variables with the values that correspond to these identifiers.
  - "pivot_wider()": performs the opposite of the previous variable.

It should be emphasized that the type of data transformation you perform is closely related to the objective you are pursuing, so there will be a multitude of possibilities and functions.

Data cleansing

In the same way that we have described the data transformation process, when considering data cleansing the same principles apply. Depending on the "final" format and the specific use you want to give to the data, you will need to use the data cleansing principles and functions in one way or another.

An example to illustrate this is how to deal with unrecorded data due to errors in tools reading the data or other reasons, such as one of the player's lack of response in a questionnaire. The initial question then is what do you want to do with the data you have.

Case 1: There is no data on a player in the MD-3 session. Although they participated in the training session, their GPS device did not record any activity. If at the end of the week you want to compare the difference between the current weekly volume and that of the last month, having an unlogged session could mean you would underestimate the values of this last week, so you might choose to assign values to the unlogged session. There are different reasonings and options for using one type of assignment or another, but you may decide to use the average of your other MD-3 sessions throughout the season.

Case 2: There is no data on a player in the MD-3 session. Although they participated in the training session, their GPS device did not record any activity. If at the end of the session you want to have a group summary of the conditional manifestation of the team in certain variables, you will omit that there is a player for whom data has not been collected, since if you assigned values "0" to that player, the data would not be representing the reality of the training session because the player did participate in it.

Case 3: A player does not have data for the MD-3 session because they did not participate due to physical discomfort. If you want to perform the same analysis as in case 1, in this case you should choose to indicate that all the values for that day are "0", in order to accurately record in your calculations that there was no activity that day.

This example serves to show the specificity of each of the cases with which you may work at some point. The functions of RStudio (Allaire, 2021) will only make it easier for you to get the desired result. As in the previous step, these are the main features used during the data cleansing process.

- Missing data
  - "na.omit()" function: in "R", if there is incomplete data in any of the columns, it is recorded as "NA". This feature allows you to remove them. The "complete.cases()" function aims to achieve the same goal but by selecting only the columns that have all the complete observations.
  - “is.na()" function: it identifies observations where data is missing.
- Extreme data
  - You should use functions that describe the data so that you can detect values that are outside of acceptable ranges and filter them accordingly.
- Cleaning columns with text
  - “clean_names()" function: it names columns in a simple format that allows for an efficient use.
  - “gsub()" function: it detects patterns in the text of the observations and replaces it with the desired format.

Self-assessment

Depending on the "final" format and the specific use you want to give to the data, you will need to use the data cleansing principles and functions in one way or another.

Depending on the "interim" format you want to give to the data and the general use, you should use the data cleansing principles and functions in a unique way in all cases.

SUBMIT

CONTINUE

References

Allaire, J. J. (2021). *RStudio* (09.0). [entorno de desarrollo integrado para el lenguaje de programación]. Posit. https://posit.co/products/open-source/rstudio-server/

Broman, K. W. and Woo, K. H. (2018). Data Organization in Spreadsheets. *The American Statistician*, 72(1), pp. 2-10. https://doi.org/10.1080/00031305.2017.1375989

Grolemund, G. and Wickham, H. (2017). R for Data Science. O'Reilly Media.

CONTINUE