How Regexes in Power BI using Python and R Can Save Your Life in Extreme Cases

Luca Zavarella

Sponsors

altitudo

beanTech
IT moves your business

fluentis
the ERP ready to go live!

QUANTUMDATIS

[stesi]
Powered by Innovation

With the support of:

CREDEM CREDEMTEL

‹packt›

DATA SATURDAYS

ìnn0va community

About me

Luca Zavarella

Working in Business Intelligence with SQL Server since 2007

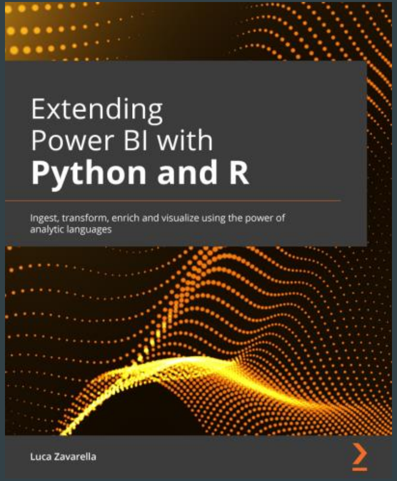
Microsoft MVP for Artificial Intelligence & Data Platform

Microsoft Certified: Azure Data Scientist Associate

Author of the book "*Extending Power BI with Python and R*" published by Packt

Head of Data & AI @ icubed

Email:	luca.zavarella@icubed.it
Twitter:	@lucazav
LinkedIn:	https://it.linkedin.com/in/lucazavarella
Blog:	https://lucazavarella.medium.com


Extending
Power BI with
Python and R
Ingest, transform, enrich and visualize using the power of analytic languages
Luca Zavarella

DATA SATURDAYS

inn0va community

Agenda

- What is a Regex
- Basics of Regex
- How To Configure Python and R in Power BI
- Case 1: Validating Emails and Dates in Power BI
  - Validating Emails and Dates with Regex
  - Demo 1
- Case 2: Parsing Free Text Notes in Power BI
  - Parsing Free Text Notes with Regex
  - Demo 2

With the support of:

Survey

How many of you are familiar with regular expressions (regex)?

How many of you know Python and/or R?

How many of you read my book “*Extending Power BI with Python and R*”?

What Is a Regex

Not only a bunch of characters at random

With the support of:

DATA SATURDAYS

Definition of Regex

A regular expression (or simply regex) is a generalized way to match patterns with sequences of characters (*abstract search pattern*)

Regular expressions are a *mathematical technique* originated in 1951 by experts in formal language and theoretical computer science

With the support of:

DATA SATURDAYS

Regex in Practice

Find & Replace Specific Strings

Extract substrings of a text that follows a specific pattern, and eventually replace them

Data Validation

Email, dates, phone numbers, credit card validations

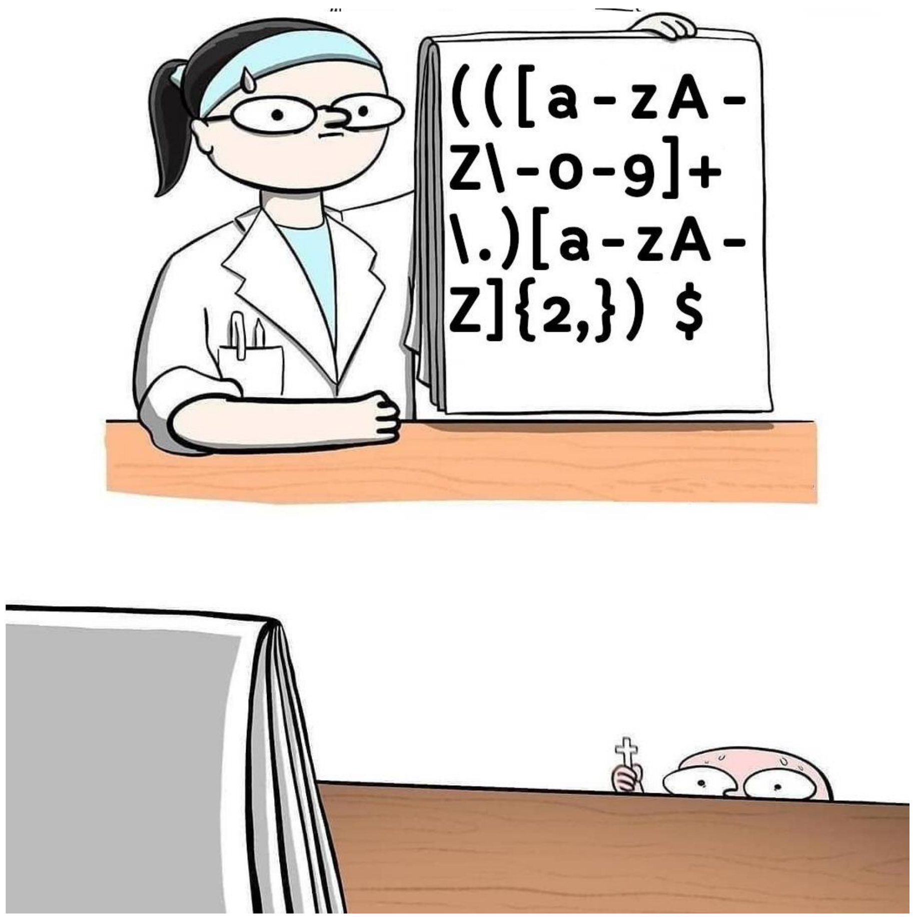
Password Pattern Matching

“Passwords must have at least 8 characters and contain at least two of the following: uppercase letters, lowercase letters, numbers, and symbols”

Syntax Highlighting

Emacs's syntax highlighting and indentation are implemented almost exclusively with regexes

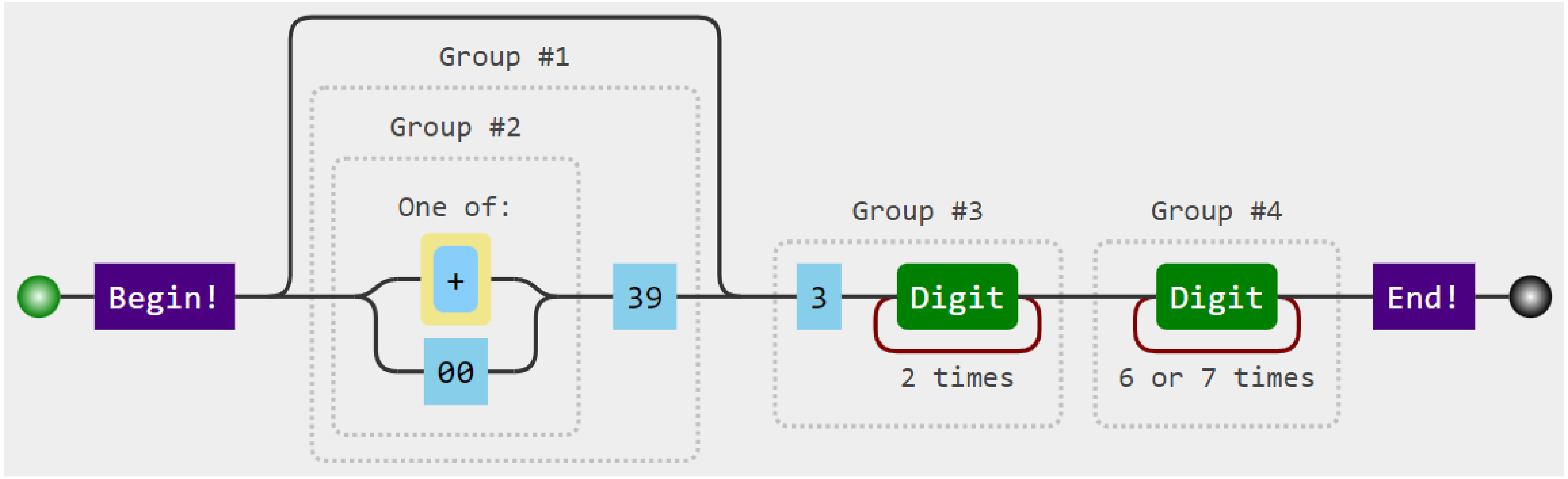
How a Regex Looks Like



With the support of:

A Simple Example of Regex

```
/^(([+]|00)39)?(3\d{2})(\d{6,7})$/g
```



https://jex.im/regulex

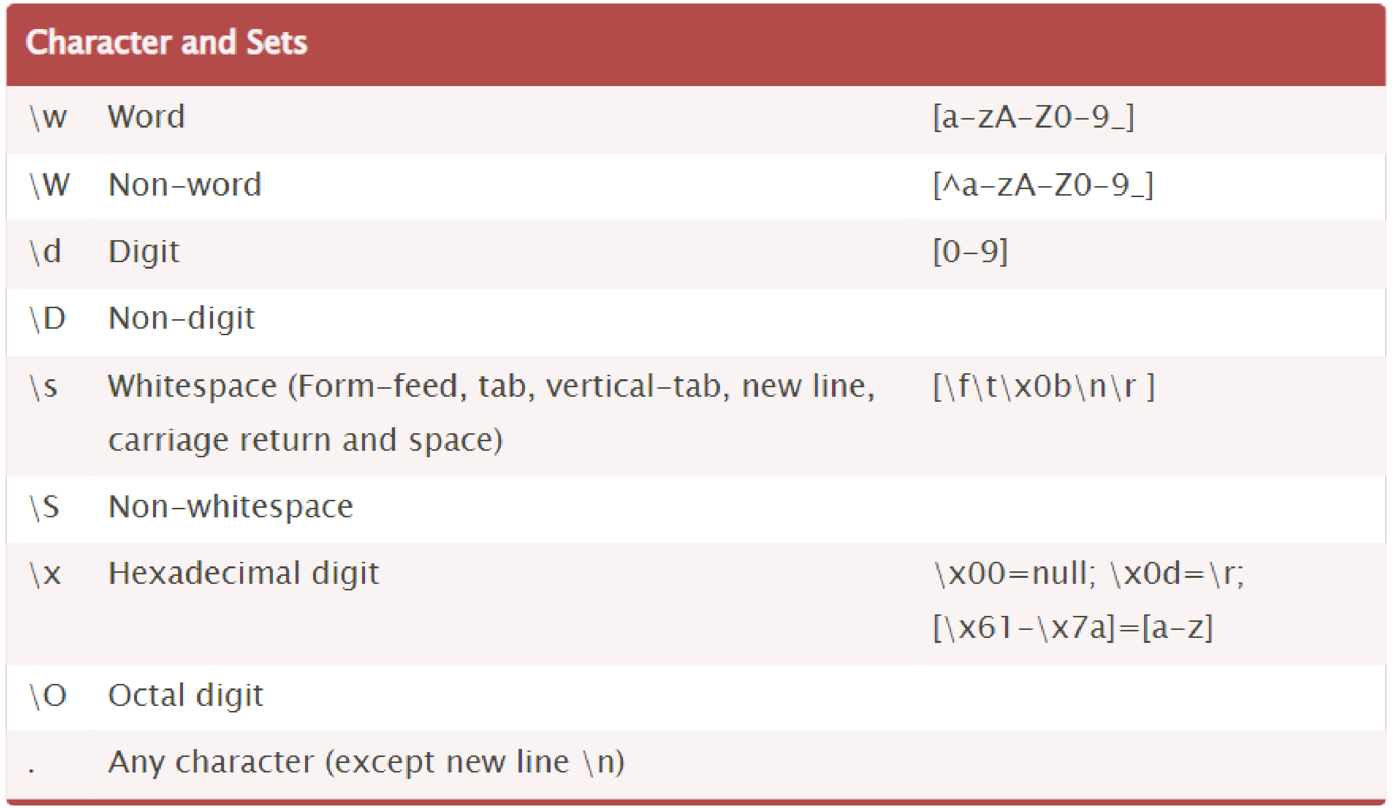
With the support of:

Basics of Regex

Let's get in touch with the core concepts

With the support of:

Character and Sets

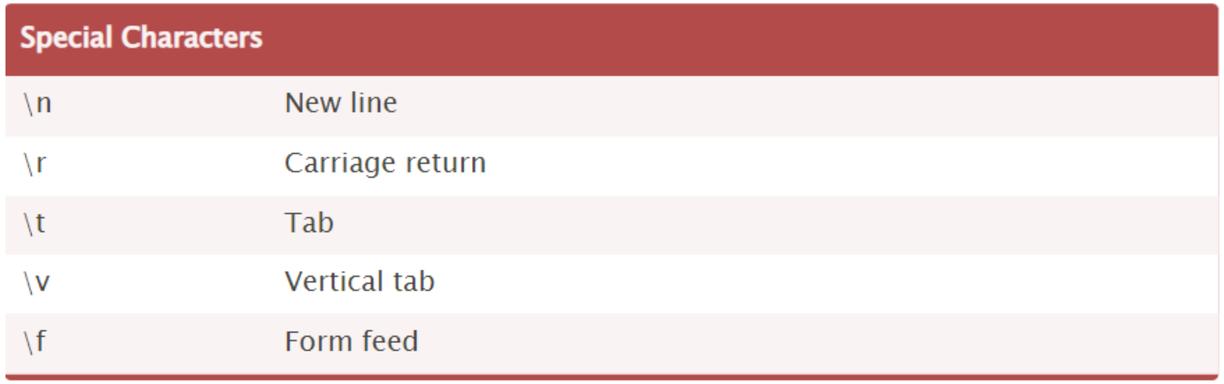


Character and Sets

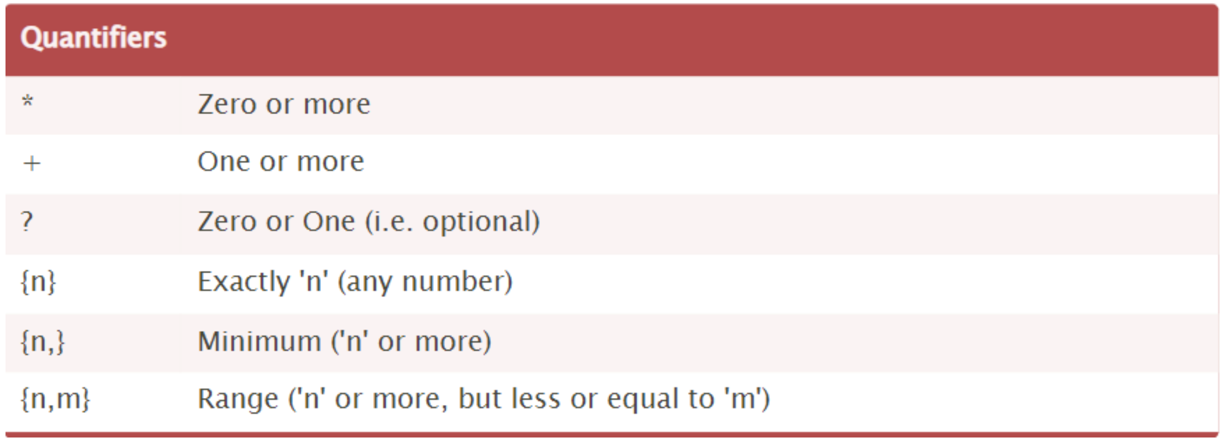
\w	Word	[a-zA-Z0-9_]
\W	Non-word	[^a-zA-Z0-9_]
\d	Digit	[0-9]
\D	Non-digit	
\s	Whitespace (Form-feed, tab, vertical-tab, new line, carriage return and space)	[\f\t\x0b\n\r]
\S	Non-whitespace	
\x	Hexadecimal digit	\x00=null; \x0d=\r; [\x61-\x7a]=[a-z]
\O	Octal digit	
.	Any character (except new line \n)	

With the support of:

Special Characters and Quantifiers



Special Characters	
\n	New line
\r	Carriage return
\t	Tab
\v	Vertical tab
\f	Form feed



Quantifiers	
*	Zero or more
+	One or more
?	Zero or One (i.e. optional)
{n}	Exactly 'n' (any number)
{n,}	Minimum ('n' or more)
{n,m}	Range ('n' or more, but less or equal to 'm')

With the support of:

Greedy and Lazy Quantifiers 1/2

By default, quantifiers are greedy!

Regex: `\d+` → 12345abc678-def (2 matches)

The question mark ? makes quantifiers lazy

Regex: `\d+?` → 12345abc678-def (8 matches)

Greedy and Lazy Quantifiers 2/2

Another example of greedy versus lazy quantifiers:

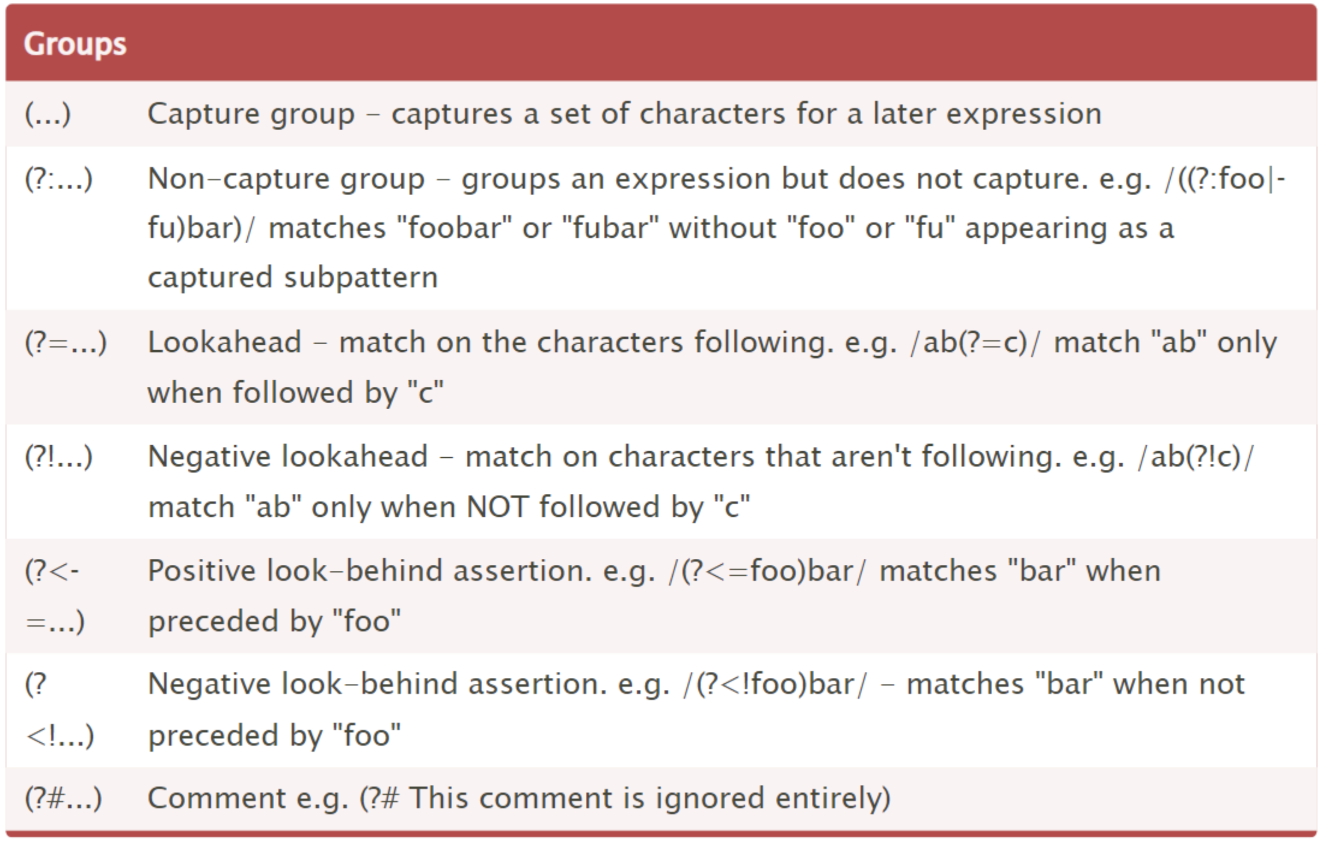
Regex: 3.*\d ➡ 123EEE2345 (1 matches)

The question mark ? makes quantifiers lazy

Regex: 3.*?\d ➡ 123EEE2345 (2 matches)

With the support of:

Groups and Lookarounds



Groups	
(...)	Capture group – captures a set of characters for a later expression
(?:...)	Non-capture group – groups an expression but does not capture. e.g. /((?:foo|-fu)bar)/ matches "foobar" or "fubar" without "foo" or "fu" appearing as a captured subpattern
(?=...)	Lookahead – match on the characters following. e.g. /ab(?=c)/ match "ab" only when followed by "c"
(?!...)	Negative lookahead – match on characters that aren't following. e.g. /ab(?!c)/ match "ab" only when NOT followed by "c"
(?<-=...)	Positive look-behind assertion. e.g. /(?<=foo)bar/ matches "bar" when preceded by "foo"
(?<!...)	Negative look-behind assertion. e.g. /(?<!foo)bar/ – matches "bar" when not preceded by "foo"
(?#...)	Comment e.g. (?# This comment is ignored entirely)

With the support of:

DATA SATURDAYS

Don't Underestimate Those Who Knows Regex

With the support of:

Power BI And Regular Expressions

Unfortunately, Power BI doesn't handle regex out-of-the-box

For this reason, we will resort to Python and R in Power BI!

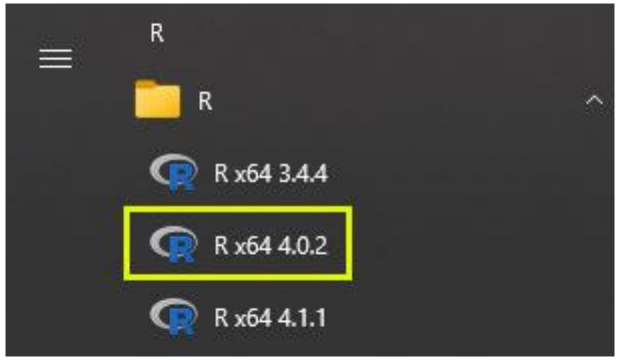
How To Configure Python and R in Power BI

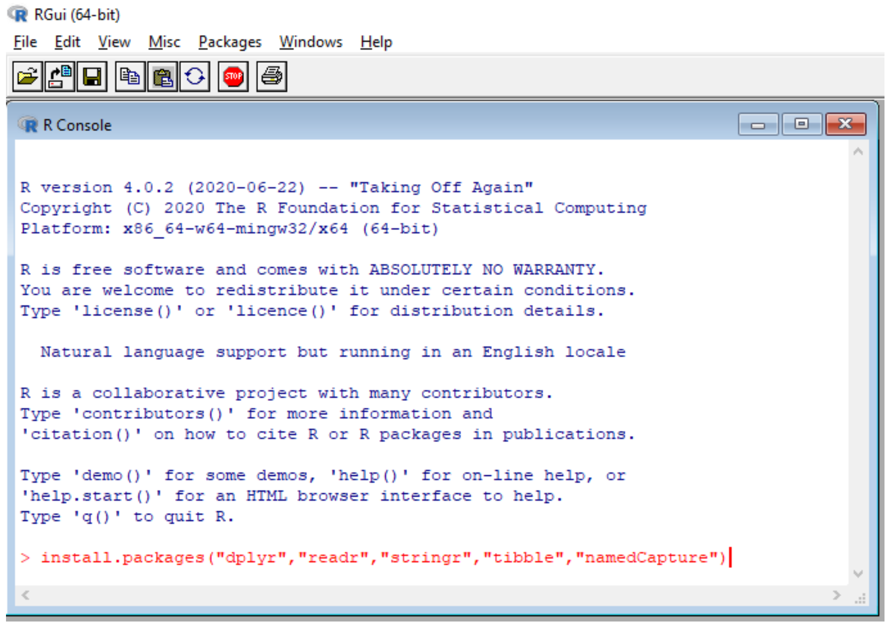
Let's oil the engine before setting off at full speed

With the support of:

Let's Configure R…

1. Download and install CRAN R
   - https://cran.r-project.org/
2. Install the required packages (*dplyr, readr, stringr, tibble, namedCapture, readxl*):



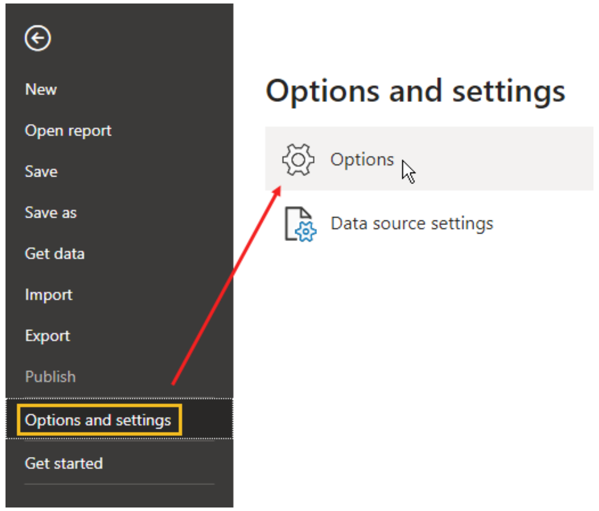


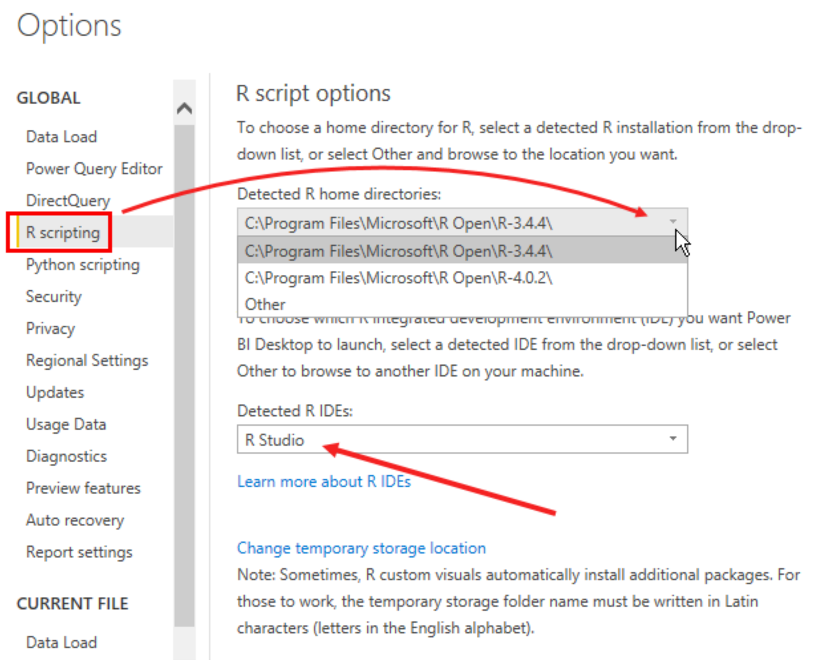
```
R version 4.0.2 (2020-06-22) -- "Taking Off Again"
Copyright (C) 2020 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> install.packages("dplyr","readr","stringr","tibble","namedCapture")
```

With the support of:

…Then Configure R With Power BI

1. Configure Power BI Desktop to work with R





R script options

To choose a home directory for R, select a detected R installation from the drop-down list, or select Other and browse to the location you want.

Detected R home directories:

C:\Program Files\Microsoft\R Open\R-3.4.4\
C:\Program Files\Microsoft\R Open\R-3.4.4\
C:\Program Files\Microsoft\R Open\R-4.0.2\
Other

To choose which R integrated development environment (IDE) you want Power BI Desktop to launch, select a detected IDE from the drop-down list, or select Other to browse to another IDE on your machine.

Detected R IDEs:

R Studio

Learn more about R IDEs

Change temporary storage location

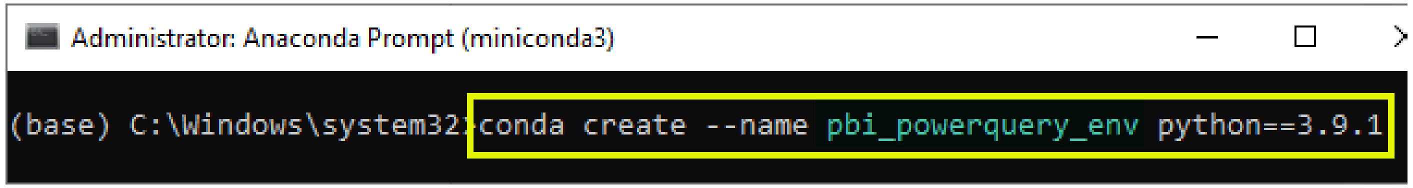
Note: Sometimes, R custom visuals automatically install additional packages. For those to work, the temporary storage folder name must be written in Latin characters (letters in the English alphabet).

With the support of:

DATA SATURDAYS

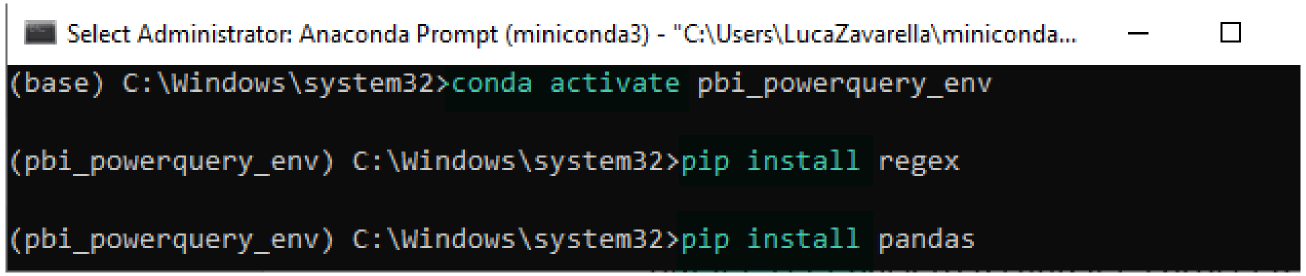
Let’s Configure Python…

1. Download and install MiniConda (*https://docs.conda.io/en/latest/miniconda.html*)
2. Use the Anaconda Prompt to create a dedicated Conda Environment with the chosen Python version:


Administrator: Anaconda Prompt (miniconda3)

```
(base) C:\Windows\system32>conda create --name pbi_powerquery_env python==3.9.1
```

3. Install the required packages (*regex, pandas, openpyxl*):

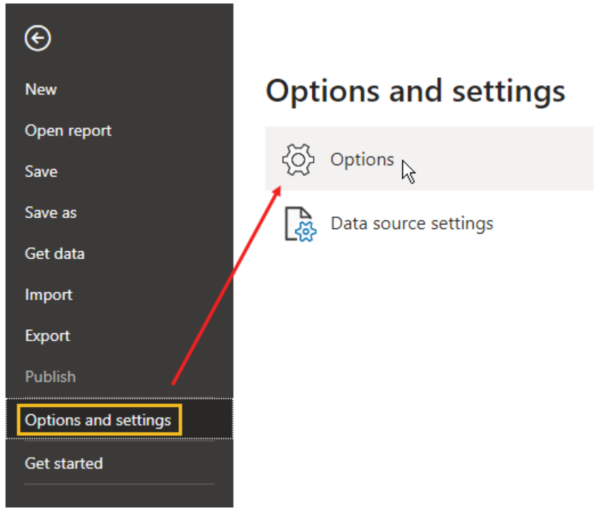

Select Administrator: Anaconda Prompt (miniconda3) - "C:\Users\LucaZavarella\miniconda...

```
(base) C:\Windows\system32>conda activate pbi_powerquery_env

(pbi_powerquery_env) C:\Windows\system32>pip install regex

(pbi_powerquery_env) C:\Windows\system32>pip install pandas
```

With the support of:

Configure Python with Power BI

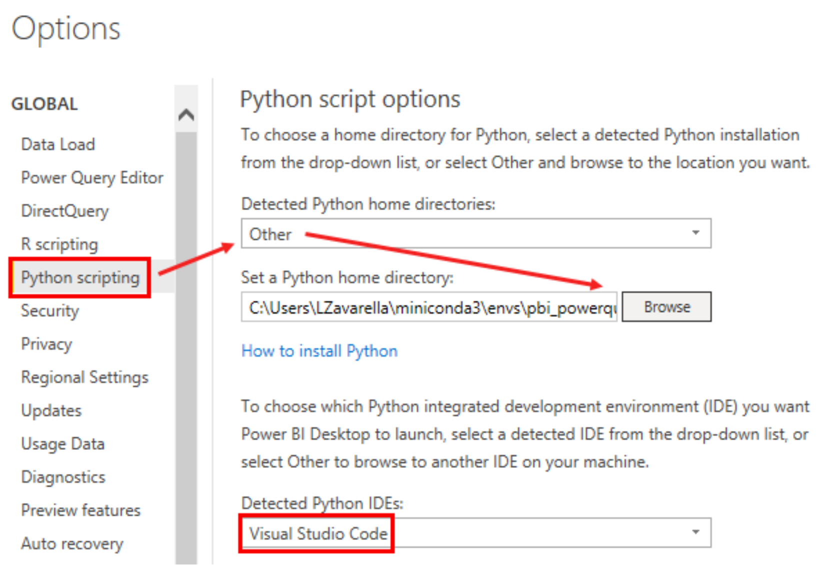
1. Configure Power BI Desktop to work with your Python Environment



New
Open report
Save
Save as
Get data
Import
Export
Publish
Options and settings
Get started

Options and settings

Options
Data source settings



Options

GLOBAL
Data Load
Power Query Editor
DirectQuery
R scripting
Python scripting
Security
Privacy
Regional Settings
Updates
Usage Data
Diagnostics
Preview features
Auto recovery

Python script options

To choose a home directory for Python, select a detected Python installation from the drop-down list, or select Other and browse to the location you want.

Detected Python home directories: Other

Set a Python home directory: C:\Users\LZavarella\miniconda3\envs\pbi_powerqu Browse

How to install Python

To choose which Python integrated development environment (IDE) you want Power BI Desktop to launch, select a detected IDE from the drop-down list, or select Other to browse to another IDE on your machine.

Detected Python IDEs: Visual Studio Code

With the support of:

CASE 1: Validating Emails and Dates

How to bring attention to a possible human error

With the support of:

Case 1 Description

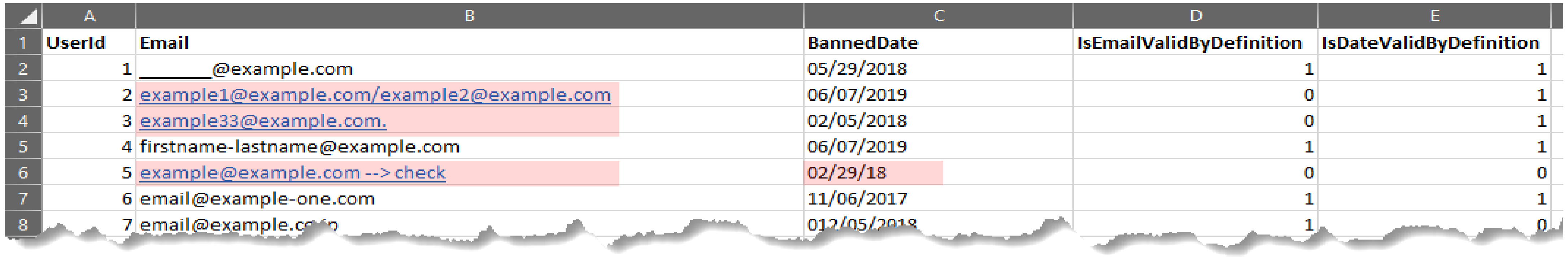
- In a retail company, a team is dedicated to identifying fraudulent customers
- The team fills out an Excel spreadsheet, in which the "*Email*" and "*BannedDate*" information of the fraudster is included

Goal

Select from other data sources only the fraudsters' information to analyze their purchases in Power BI

What Happens During Excel Completion

Unfortunately, sometimes typos can happen during the data entry...



	A	B	C	D	E
1	UserId	Email	BannedDate	IsEmailValidByDefinition	IsDateValidByDefinition
2	1	________@example.com	05/29/2018	1	1
3	2	example1@example.com/example2@example.com	06/07/2019	0	1
4	3	example33@example.com.	02/05/2018	0	1
5	4	firstname-lastname@example.com	06/07/2019	1	1
6	5	example@example.com --> check	02/29/18	0	0
7	6	email@example-one.com	11/06/2017	1	1
8	7	email@example.c	012/05/2018	1	0

We need to identify any errors and highlighting them, allowing the fraud team to be able to correct them

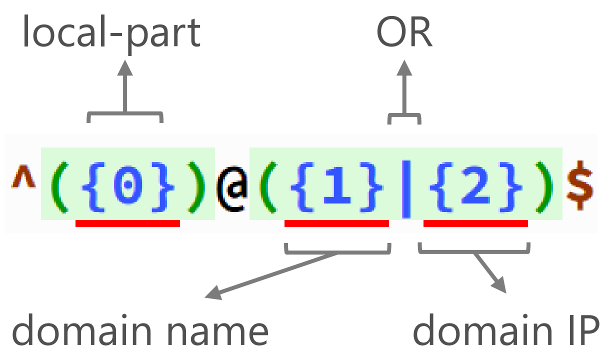
With the support of:

Format of an Email Address

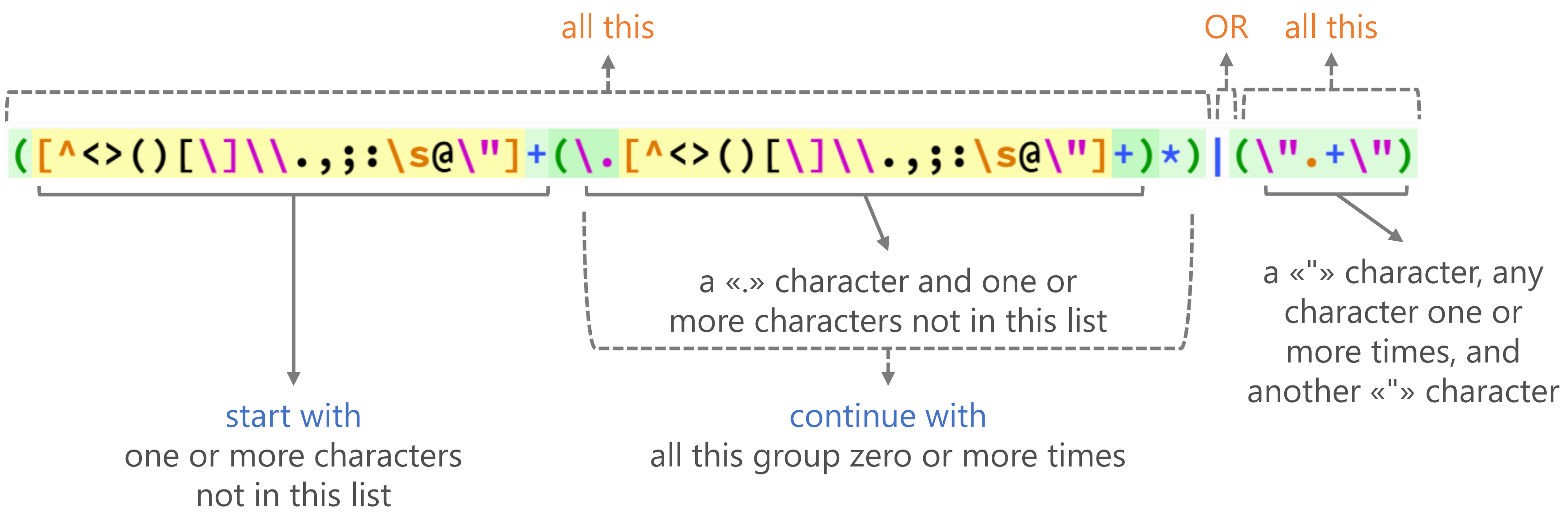
Generic format of an email address:

local-part@domain

where “domain” can be a domain name or a domain IP. In a “regex point of view”:



Validating the Local Part of Email



```
([^<>()[\]\\.,;:\s@\"]+(\.[^<>()[\]\\.,;:\s@\"]+)*)|(\".+\")
```

all this

OR

all this

a «.» character and one or more characters not in this list

a «"» character, any character one or more times, and another «"» character

start with
one or more characters not in this list

continue with
all this group zero or more times

With the support of:

DATA SATURDAYS

Validating the Domain of an Email

Domain name:

start with
all this group one or
more times

continue with
a letter character two or
more times

```
(([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,})
```

one or more alphanumeric
characters, including the dash
«-», and the «.» character

Domain IP:

an optional
«[» character

a «.» character

a «.» character

a «.» character

an optional
«]» character

```
(\[?[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\]?)
```

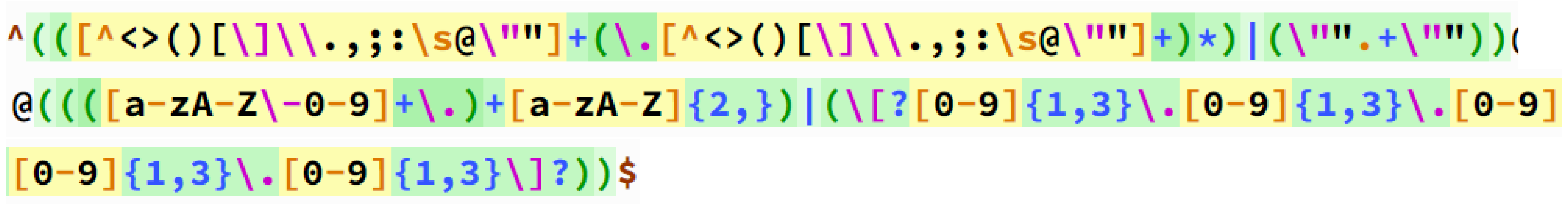
one of digit
charachters one
to three times

one of digit
charachters one
to three times

one of digit
charachters one
to three times

one of digit
charachters one
to three times

Final Regex for Email Validation



```
^(([^<>()[\]\\.,;:\s@\""]+(\.[^<>()[\]\\.,;:\s@\""]+)*)|(\"".+\""))(
@((([a-zA-Z\-0-9]+\.)+[a-zA-Z]{2,})|(\[?[0-9]{1,3}\.[0-9]{1,3}\.[0-9]
[0-9]{1,3}\.[0-9]{1,3}\]?))$
```

With the support of:

How It Should Be According to RFC822

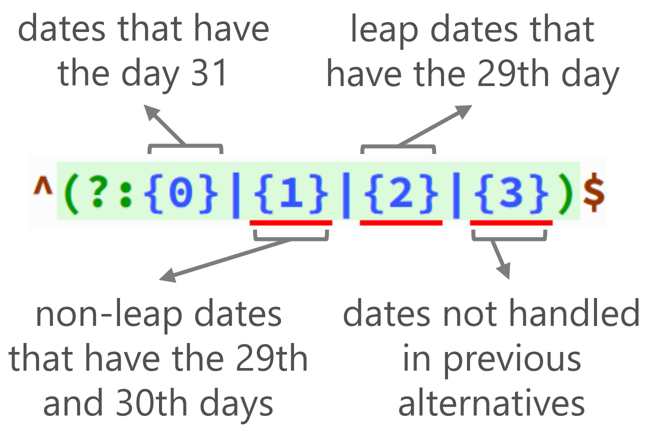
```
(?:(?:\r\n)?[ \t])*(?:(?:(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,; [illegible] [\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:\r\n)?[
\t])*)(?:\.(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["( [illegible] [\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:\r\n)?[
\t])*))*@(?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["( [illegible] ]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ \t])*)(?:\.(?:(?:\r\n)?[
\t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["() [illegible] \\".\[\]]))|\[( [illegible] *\](?:(?:\r\n)?[ \t])*))*|(?:[^()<>@,;:\\".\[\]
\000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\ [illegible] r\n)?[ \t])*)*\<(?:(?:\r\n)?[ \t])*(?:@(?:[^()<>@,;:\\".\[\]
\000-\031]+(?:(?:(?:\r\n)?[ \t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([ [illegible] (?:(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)? [illegible] ()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ [illegible] <>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)? [illegible] [^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]) [illegible] ?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*" [illegible] \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ [illegible] ".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ [illegible] ".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?: [illegible] (?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|( [illegible] :[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|( [illegible] [^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.) [illegible] .\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.) [illegible] (?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?: [illegible] (?:@(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] ".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] \[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] \\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?: [illegible] *(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n) [illegible] (?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n [illegible] ".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r [illegible] ()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]) [illegible] \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n) [illegible] *(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:( [illegible] n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r [illegible] ^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] 000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?: [illegible] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ \t])* [illegible] 000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ \t])*))*)*: [illegible] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:\r\n)?[ [illegible] *(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|"(?:[^\"\r\\]|\\.|(?:(?:\r\n)?[ \t]))*"(?:(?:\r\n)?[ [illegible] :(?:\r\n)?[ \t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ \t])*)(?:\.(?:(?: [illegible] t])*(?:[^()<>@,;:\\".\[\] \000-\031]+(?:(?:(?:\r\n)?[
\t])+|\Z|(?=[\["()<>@,;:\\".\[\]]))|\[([^\[\]\r\\]|\\.)*\](?:(?:\r\n)?[ \t])*))*\>(?:(?:\r\n)?[ \t])*))*)?;\s*)
```

With the support of:

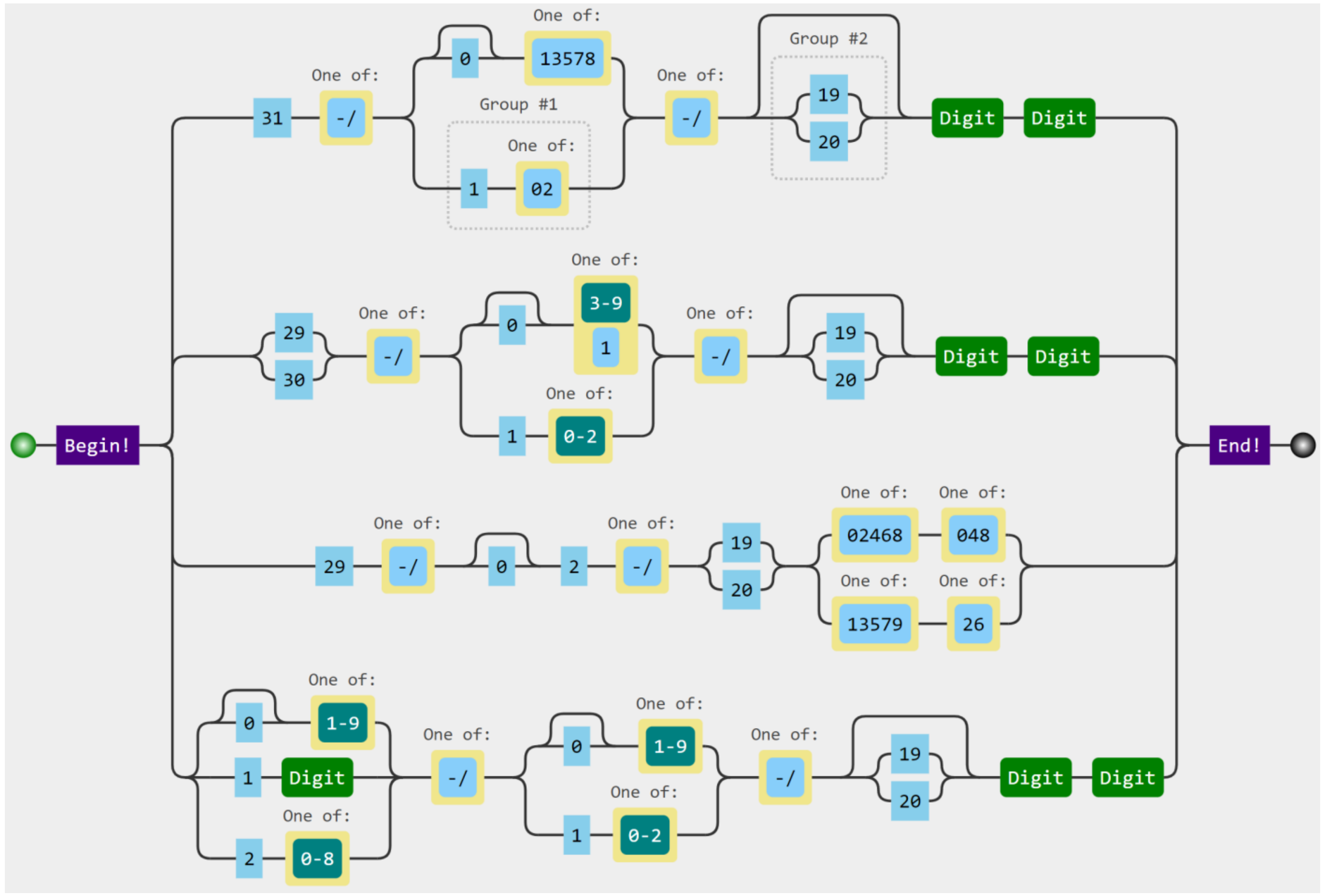
Format of a Date

Dates can be partitioned as following to validate them also "semantically":

- Dates having the day 31
- Non-leap dates having the 29th and 30th days
- Leap dates having the 29th day
- Dates not handled in previous cases


dates that have the day 31
leap dates that have the 29th day
^(?:{0}|{1}|{2}|{3})$
non-leap dates that have the 29th and 30th days
dates not handled in previous alternatives

Validating a Date



With the support of:

Demo 1

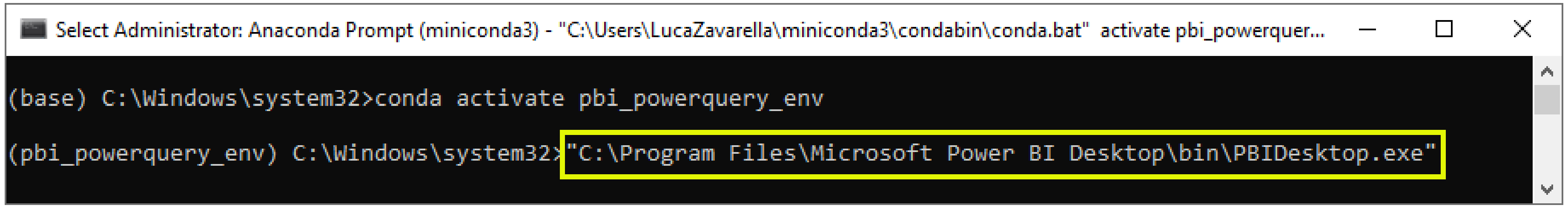
DEMO 1

Validating Emails and Dates in Power BI

How to properly validate information entered by users

Be Aware To the ADO.NET Python Script Error

- Executing a Python script in Power BI you may run into the "*ADO.NET Python error*" (not the real error!)
- In my case the error is "*Unable to import required dependencies: numpy*"
- Anaconda requires the environment to be activated
- Power BI Desktop will directly invoke `python.exe` which doesn't have an initialized environment out of the box
- The solution is launching the Power BI Desktop executable from the Anaconda Prompt after activating the proper environment


Select Administrator: Anaconda Prompt (miniconda3) - "C:\Users\LucaZavarella\miniconda3\condabin\conda.bat" activate pbi_powerquer...

```
(base) C:\Windows\system32>conda activate pbi_powerquery_env

(pbi_powerquery_env) C:\Windows\system32>"C:\Program Files\Microsoft Power BI Desktop\bin\PBIDesktop.exe"
```

With the support of:

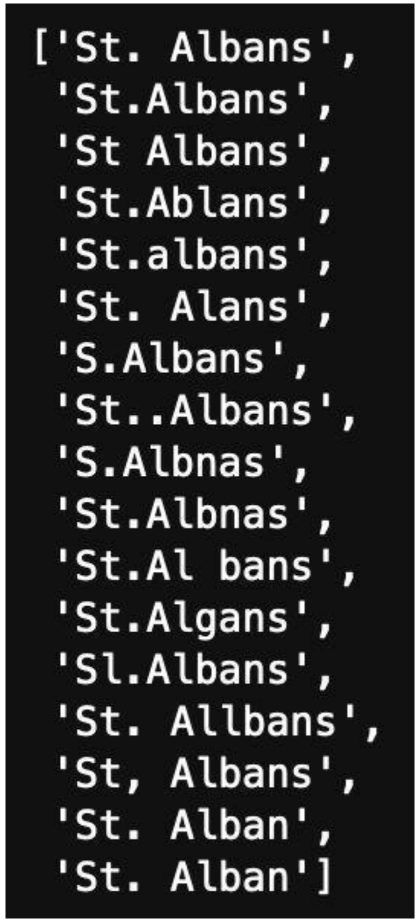
CASE 2: Parsing Free Text Notes

When the human's imagination exceeds all expectations

With the support of:

“We have incredibly valuable datasets”

The dataset…



```
['St. Albans',
 'St.Albans',
 'St Albans',
 'St.Ablans',
 'St.albans',
 'St. Alans',
 'S.Albans',
 'St..Albans',
 'S.Albnas',
 'St.Albnas',
 'St.Al bans',
 'St.Algans',
 'Sl.Albans',
 'St. Allbans',
 'St, Albans',
 'St. Alban',
 'St. Alban']
```

With the support of:

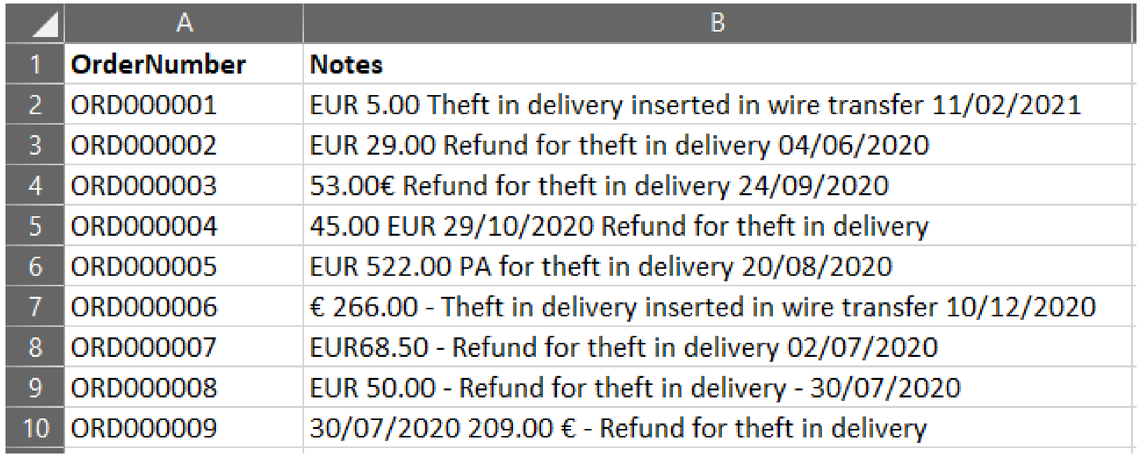
The Nightmare of Analysts: Free Text Notes

Sometimes a fraudster manages to steal goods addressed to a customer and therefore the customer asks to be refunded by the company

The defrauded customer contacts Customer Care to request a refund

The management system provided to the Customer Care operator doesn't allow to enter and validate the information of the refund in a structured way

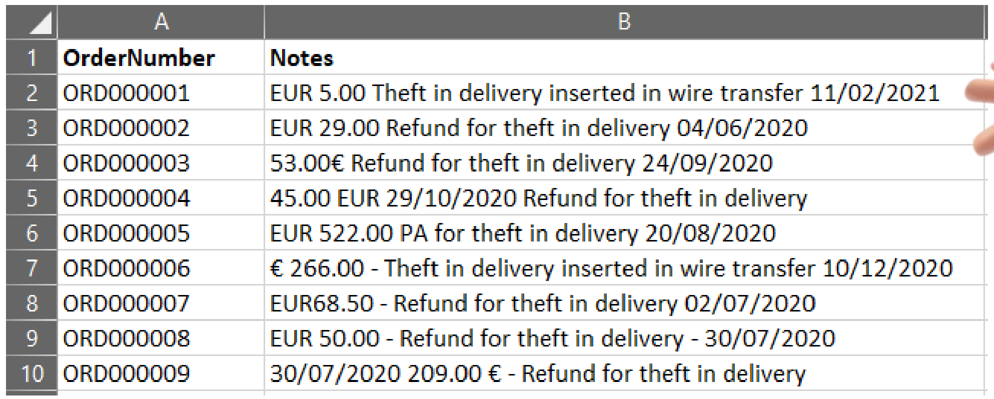
The operator have to resort to the only possible method: the entry of a free text note for the order!



	A	B
1	**OrderNumber**	**Notes**
2	ORD000001	EUR 5.00 Theft in delivery inserted in wire transfer 11/02/2021
3	ORD000002	EUR 29.00 Refund for theft in delivery 04/06/2020
4	ORD000003	53.00€ Refund for theft in delivery 24/09/2020
5	ORD000004	45.00 EUR 29/10/2020 Refund for theft in delivery
6	ORD000005	EUR 522.00 PA for theft in delivery 20/08/2020
7	ORD000006	€ 266.00 - Theft in delivery inserted in wire transfer 10/12/2020
8	ORD000007	EUR68.50 - Refund for theft in delivery 02/07/2020
9	ORD000008	EUR 50.00 - Refund for theft in delivery - 30/07/2020
10	ORD000009	30/07/2020 209.00 € - Refund for theft in delivery

DATA SATURDAYS

It Will Definitely Arrive That Day…



	A	B
1	**OrderNumber**	**Notes**
2	ORD000001	EUR 5.00 Theft in delivery inserted in wire transfer 11/02/2021
3	ORD000002	EUR 29.00 Refund for theft in delivery 04/06/2020
4	ORD000003	53.00€ Refund for theft in delivery 24/09/2020
5	ORD000004	45.00 EUR 29/10/2020 Refund for theft in delivery
6	ORD000005	EUR 522.00 PA for theft in delivery 20/08/2020
7	ORD000006	€ 266.00 - Theft in delivery inserted in wire transfer 10/12/2020
8	ORD000007	EUR68.50 - Refund for theft in delivery 02/07/2020
9	ORD000008	EUR 50.00 - Refund for theft in delivery - 30/07/2020
10	ORD000009	30/07/2020 209.00 € - Refund for theft in delivery

With the support of:

Entities to Extract From Free Text

Refund amount | Refund reason | Refund date

Refund amount made by *currency* and *amount*

*E*ntered as: “EUR xx.yy”, “EURxx.yy”, “xx.yy EUR”, “€ xx.yy”, “xx.yy€”, “xx.yy €”, etc.

“Separator” between all the information can be made by one or more *white spaces* or by a *dash* surrounded by one or more spaces

Refund date is always in the *dd/mm/yyyy* format (you are lucky here! ☺)

Refund reason could contain any text

Currency:	`(?:EUR|€)`
Amount:	`\d{1,}\.?\d{0,2}`
Separator:	`(?:\s+)?-?(?:\s+)`
Date:	`\d{2}[\-\/]\d{2}[\-\/]\d{4}`
Reason:	`.*?`

One Regex to Rule Them All

```
^(?:
({currency}{separator}{amount}{separator}{reason}{separator}{date})
OR
({amount}{separator}{currency}{separator}{reason}{separator}{date})
OR
({date}{separator}{currency}{separator}{amount}{separator}{reason})
OR
({date}{separator}{amount}{separator}{currency}{separator}{reason})
)$
```

With the support of:

DATA SATURDAYS

Demo 2

Parsing Free Text Notes in Power BI

How to get useful information from free text nots

With the support of:

References

1. Fixing ADO.NET error trying to run Python Script in Power BI | by Luca Zavarella | Microsoft Azure | Medium
2. https://www.amazon.it/Extending-Power-Python-transform-analytical/dp/1801078203

THANK YOU !!

With the support of: