Proyecto de Transformación Digital de Adalab

ADALAB

+

Equipo Totoro

Adalab ReNew

Adalab ya puede **conocer e interpretar sus propios datos** y tomar decisiones basadas en hechos.

Las empresas que han hecho este cambio han aumentado su **productividad, eficiencia y resultados:**

- Minimizar riesgos
- Identificar oportunidades
- Reducir los tiempos

¡**Adalab** ya está lista para comenzar su **transformación digital**!

¿Cómo?

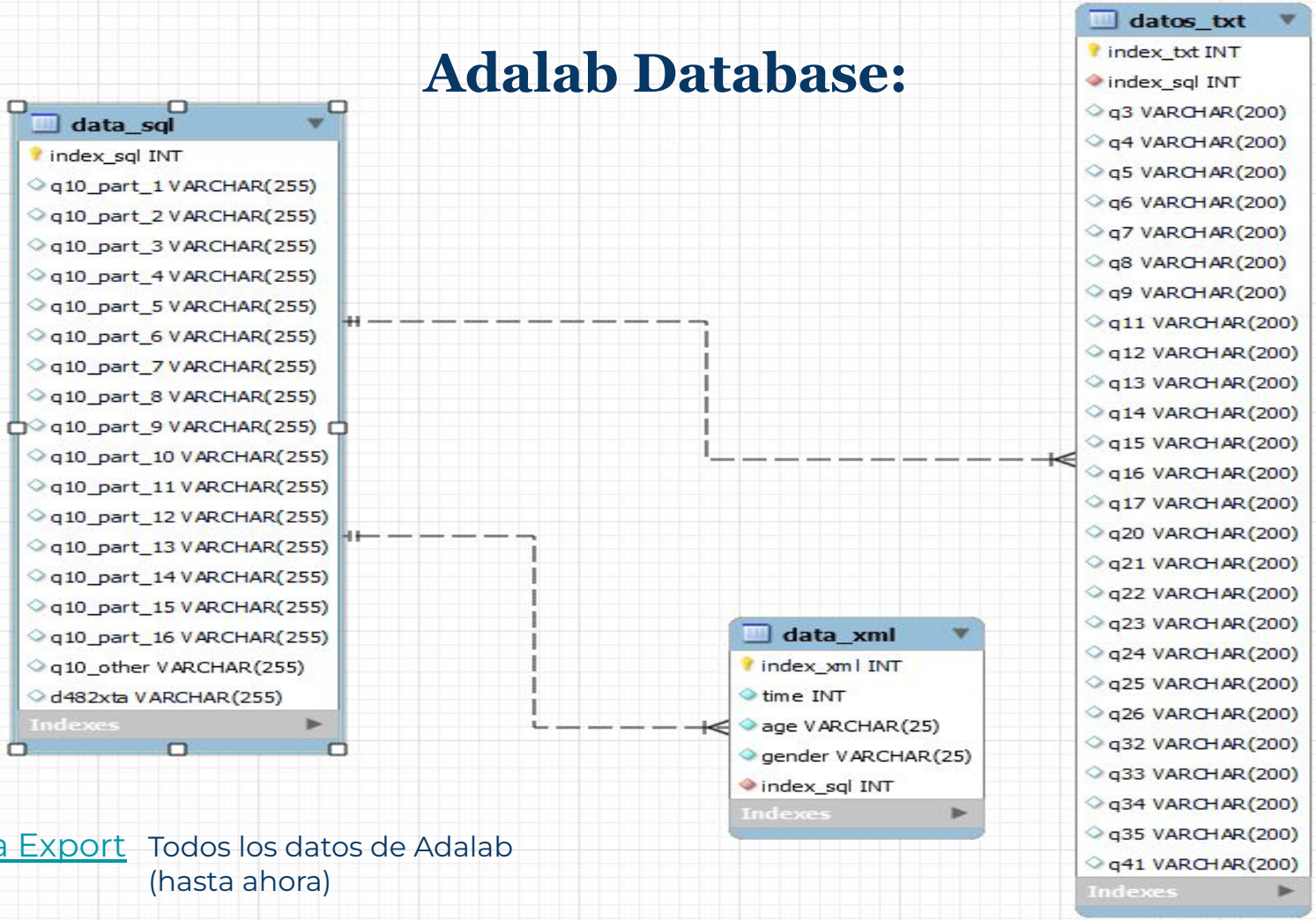
¿Cómo empezamos?

Para saber qué dicen los datos, debemos almacenarlos primero.

Para ello hemos creado una **Base de Datos propia de Adalab**.

- Cualquiera puede acceder a los datos de forma rápida y segura.
- Va creciendo con los datos insertados periódicamente.
- Los datos pueden relacionarse entre sí para extraer conclusiones.
- Todo queda en un lugar.

Adalab Database:



Data Export Todos los datos de Adalab (hasta ahora)

Las manos en la data

Los datos que entran, lo hacen más y mejor si están limpios y organizados con la misma lógica.

Entran en formato *.txt*, *.xml* o *.sql* y se almacenan en la Base de Datos de Adalab.

El **código de procesado de datos** en *.xml* y *.txt*:

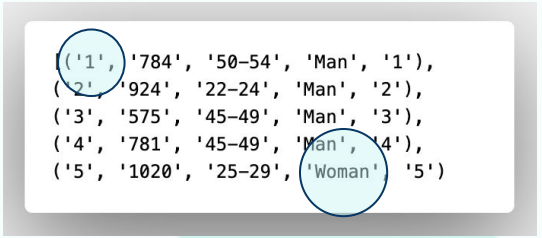
- Está almacenado en clases de funciones.
- Facilita la automatización del proceso.
- Específica para Adalab.
- Adaptable y ampliable a necesidades futuras.

Código Automatización - Clases

```
class Limpieza_txt:

    def __init__ (self, txt):
        self.txt = txt

    #definimos función txt a lista de listas
    def txt_to_list(self):
        lista_resultado = []
        for i in self.txt:
            i = i.split(';')
            lista_resultado.append(i)
        return lista_resultado

    # definimos función salto en línea

    def quitar_salto(self):
        try:
            self.txt = self.txt_to_list()

            for list in self.txt:
                for i in list:
                    if i == list[-1]:
                        i = i.rstrip()
                        list[-1] = i

            return self.txt

        except:
            for list in self.txt:
                for i in list:
                    if i == list[-1]:
                        i = i.rstrip()
                        list[-1] = i

            return self.txt
```

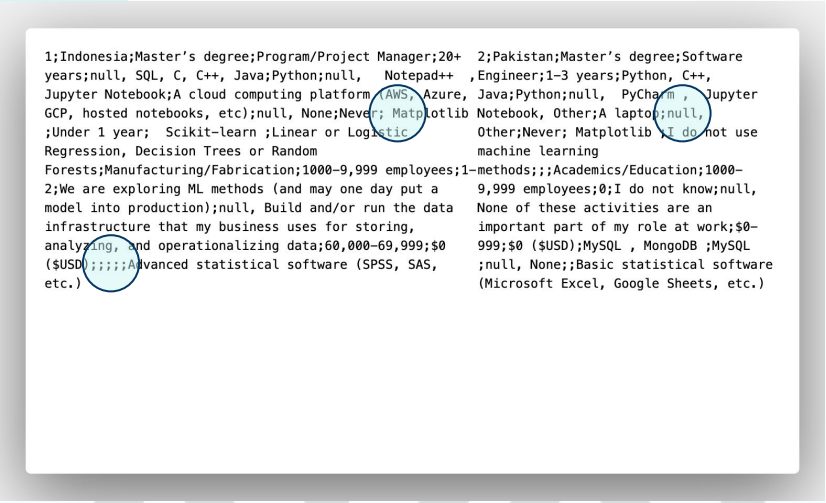
Antes

Después



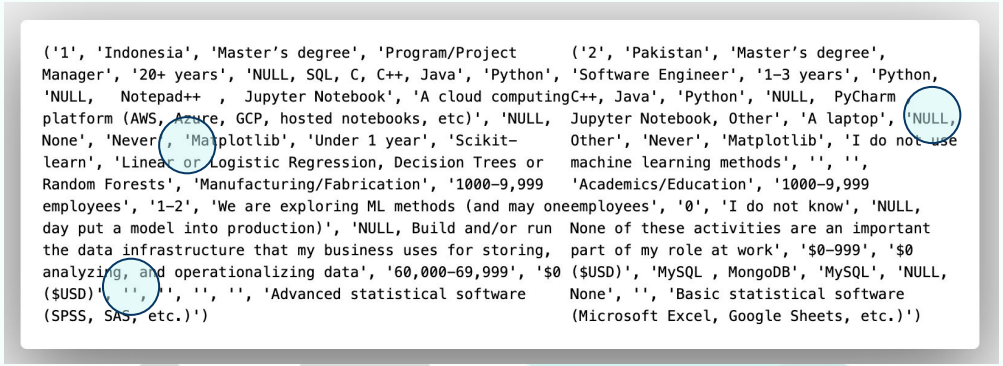
Cambios de nomenclatura
Especificación del campo “gender”
Reorganización de la info

Antes



```
1;Indonesia;Master's degree;Program/Project Manager;20+ years;null, SQL, C, C++, Java;Python;null,   Notepad++  ,Jupyter Notebook;A cloud computing platform (AWS, Azure, GCP, hosted notebooks, etc);null, None;Never; Matplotlib ;Under 1 year;  Scikit-learn ;Linear or Logistic Regression, Decision Trees or Random Forests;Manufacturing/Fabrication;1000-9,999 employees;1-2;We are exploring ML methods (and may one day put a model into production);null, Build and/or run the data infrastructure that my business uses for storing, analyzing, and operationalizing data;60,000-69,999;$0 ($USD);;;;;Advanced statistical software (SPSS, SAS, etc.)
2;Pakistan;Master's degree;Software Engineer;1-3 years;Python, C++, Java;Python;null,  PyCharm ,  Jupyter Notebook, Other;A laptop;;null, Other;Never; Matplotlib ;I do not use machine learning methods;;;Academics/Education;1000-9,999 employees;0;I do not know;null, None of these activities are an important part of my role at work;$0-999;$0 ($USD);MySQL , MongoDB ;MySQL ;null, None;;Basic statistical software (Microsoft Excel, Google Sheets, etc.)
```

Después



```
('1', 'Indonesia', 'Master's degree', 'Program/Project Manager', '20+ years', 'NULL, SQL, C, C++, Java', 'Python', 'NULL,   Notepad++  ,  Jupyter Notebook', 'A cloud computing platform (AWS, Azure, GCP, hosted notebooks, etc)', 'NULL, None', 'Never', 'Matplotlib', 'Under 1 year', 'Scikit-learn', 'Linear or Logistic Regression, Decision Trees or Random Forests', 'Manufacturing/Fabrication', '1000-9,999 employees', '1-2', 'We are exploring ML methods (and may one day put a model into production)', 'NULL, Build and/or run the data infrastructure that my business uses for storing, analyzing, and operationalizing data', '60,000-69,999', '$0 ($USD)', '', '', '', '', 'Advanced statistical software (SPSS, SAS, etc.)')
('2', 'Pakistan', 'Master's degree', 'Software Engineer', '1-3 years', 'Python, C++, Java', 'Python', 'NULL,  PyCharm  , Jupyter Notebook, Other', 'A laptop', 'NULL, Other', 'Never', 'Matplotlib', 'I do not use machine learning methods', '', '', 'Academics/Education', '1000-9,999 employees', '0', 'I do not know', 'NULL, None of these activities are an important part of my role at work', '$0-999', '$0 ($USD)', 'MySQL , MongoDB', 'MySQL', 'NULL, None', '', 'Basic statistical software (Microsoft Excel, Google Sheets, etc.)')
```

Limpieza saltos de línea y espacios
Modificación de datos (NULL, under)

Crear, Insertar y Leer Fácilmente

Los nuevos datos que entran necesitan actualizarse en la base de datos ágilmente.

El código de programación no es amigable sin una visualización que lo acompañe.

Para ello creamos **funciones de creación, inserción y lectura de datos**:

- Crear tablas nuevas
- Insertar nuevos datos
- Visualizar con DataFrames
- Todo de forma automatizada para fácil replicación.

Código Automatización - Clases

```
class Creacion_insercion:
```

```
class Lectura:

    def __init__(self, password):
        self.password = password


    def lectura_df (self, query, lista_columnas_df):
        self.query = query
        self.lista_columnas_df = lista_columnas_df

        cnx = mysql.connector.connect(user='root', password=self.password,
                                      host='127.0.0.1', database='project1')

        mycursor = cnx.cursor()
        mycursor.execute(self.query)
        myresult = mycursor.fetchall()
        df = pd.DataFrame(myresult, columns = self.lista_columnas_df)

        return print(df)
```

```
            print("Error Code:", err.errno)
            print("SQLSTATE", err.sqlstate)
            print("Message", err.msg)

        cnx.close()
```

Antes

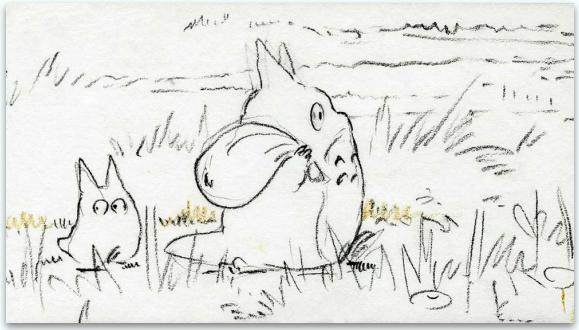
```
<data>
  <row>
    <level_0>1</level_0>
    <index>1</index>
    <time>784</time>
    <age>50-54</age>
    <gender>0</gender>
  </row>
  <row>
    <level_0>2</level_0>
    <index>2</index>
    <time>924</time>
    <age>22-24</age>
    <gender>0</gender>
  </row>
  <row>
    <level_0>3</level_0>
    <index>3</index>
    <time>575</time>
    <age>45-49</age>
    <gender>0</gender>
  </row>
  <row>
    <level_0>4</level_0>
    <index>4</index>
    <time>781</time>
    <age>45-49</age>
    <gender>0</gender>
  </row>
```

Después

Visualización por DataFrames

```
...          time    age gender  index_sql  index_xml
0             784  50-54    Man          1          1
1             924  22-24    Man          2          2
2             575  45-49    Man          3          3
3             781  45-49    Man          4          4
4            1020  25-29  Woman          5          5
...           ...    ...    ...        ...        ...
25967        1756  30-34    Man      25968      25968
25968         253  22-24    Man      25969      25969
25969         494  50-54    Man      25970      25970
25970         277  45-49    Man      25971      25971
25971         255  18-21    Man      25972      25972

[25972 rows x 5 columns]
```

Equipo Totoro



Nuestras bases...

Filosofía Agile

Aprendizaje Con el cliente

Éxito Transformación digital

Nos gusta pensar de cuántas maneras diferentes podemos ayudar con nuestro trabajo

¡Síguenos en GitHub!

Antoanela Mateciuc

Lara Sancho

Lucía Cernuda

Ariana Rosales

Chloe Aroca