Práctica de Laboratorio - Unidad 2: Introducción a la Limpieza de Datos

Introducción a la Ciencia de Datos

Unidad 2: Procesamiento y Limpieza de Datos

Título de la Práctica: Limpieza de una Base de Datos Ensuciada

Danna Patricia Riveroll Martínez

Dulce María Hernández Cervantes

Anna Yuruen Centeno Gámez

Carlos Eduardo Martínez Pérez

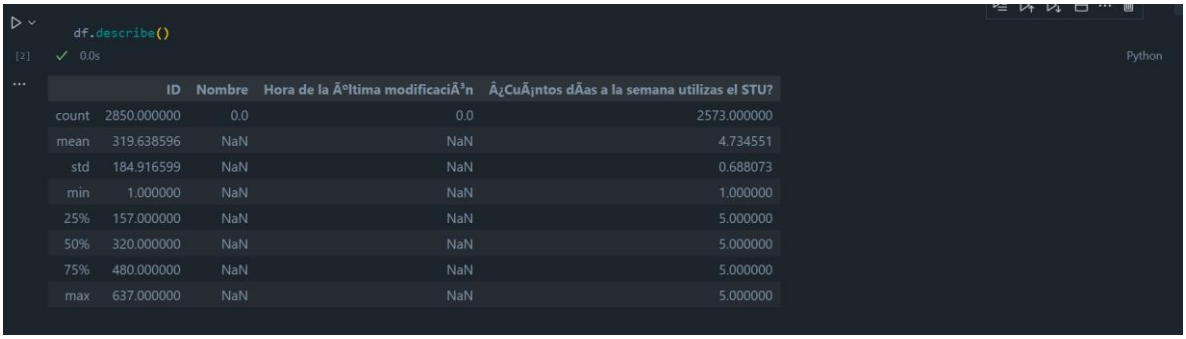
Jesús Daniel Espinosa Solano

Kevin Leonardo Marrón Ramírez

Jaime Alejandro Romero Sierra

Análisis inicial de la base de Datos

Descripción estadística de la base de datos antes de realizar la limpieza.

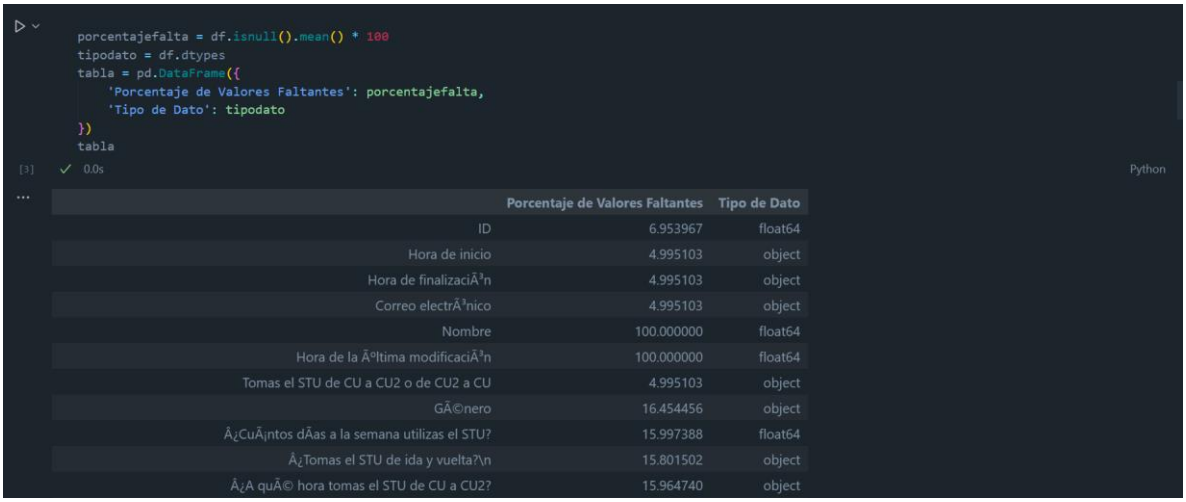


```
df.describe()
```

[2] ✓ 0.0s — Python

	ID	Nombre	Hora de la Ãºltima modificaciÃ³n	Â¿CuÃ¡ntos dÃas a la semana utilizas el STU?
count	2850.000000	0.0	0.0	2573.000000
mean	319.638596	NaN	NaN	4.734551
std	184.916599	NaN	NaN	0.688073
min	1.000000	NaN	NaN	1.000000
25%	157.000000	NaN	NaN	5.000000
50%	320.000000	NaN	NaN	5.000000
75%	480.000000	NaN	NaN	5.000000
max	637.000000	NaN	NaN	5.000000

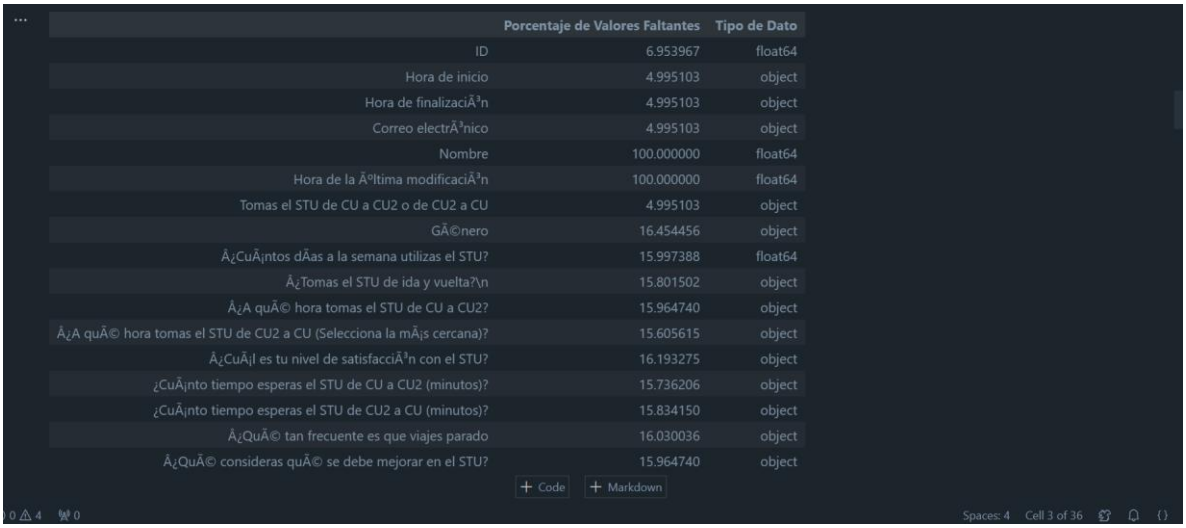
Código para general la tabla en donde se muestra el porcentaje de valore faltantes y el tipo de dato por columna.



```
porcentajefalta = df.isnull().mean() * 100
tipodato = df.dtypes
tabla = pd.DataFrame({
    'Porcentaje de Valores Faltantes': porcentajefalta,
    'Tipo de Dato': tipodato
})
tabla
```

[3] ✓ 0.0s — Python

	Porcentaje de Valores Faltantes	Tipo de Dato
ID	6.953967	float64
Hora de inicio	4.995103	object
Hora de finalizaciÃ³n	4.995103	object
Correo electrÃ³nico	4.995103	object
Nombre	100.000000	float64
Hora de la Ãºltima modificaciÃ³n	100.000000	float64
Tomas el STU de CU a CU2 o de CU2 a CU	4.995103	object
GÃ©nero	16.454456	object
Â¿CuÃ¡ntos dÃas a la semana utilizas el STU?	15.997388	float64
Â¿Tomas el STU de ida y vuelta?\n	15.801502	object
Â¿A quÃ© hora tomas el STU de CU a CU2?	15.964740	object



	Porcentaje de Valores Faltantes	Tipo de Dato
ID	6.953967	float64
Hora de inicio	4.995103	object
Hora de finalizaciÃ³n	4.995103	object
Correo electrÃ³nico	4.995103	object
Nombre	100.000000	float64
Hora de la Ãºltima modificaciÃ³n	100.000000	float64
Tomas el STU de CU a CU2 o de CU2 a CU	4.995103	object
GÃ©nero	16.454456	object
Â¿CuÃ¡ntos dÃas a la semana utilizas el STU?	15.997388	float64
Â¿Tomas el STU de ida y vuelta?\n	15.801502	object
Â¿A quÃ© hora tomas el STU de CU a CU2?	15.964740	object
Â¿A quÃ© hora tomas el STU de CU2 a CU (Selecciona la mÃ¡s cercana)?	15.605615	object
Â¿CuÃ¡l es tu nivel de satisfacciÃ³n con el STU?	16.193275	object
¿CuÃ¡nto tiempo esperas el STU de CU a CU2 (minutos)?	15.736206	object
¿CuÃ¡nto tiempo esperas el STU de CU2 a CU (minutos)?	15.834150	object
Â¿QuÃ© tan frecuente es que viajes parado	16.030036	object
Â¿QuÃ© consideras quÃ© se debe mejorar en el STU?	15.964740	object

+ Code + Markdown

Spaces: 4 Cell 3 of 36

Número de filas duplicadas.

```
df.duplicated().sum()
[4]  ✓ 0.0s                                                    Python
...  np.int64(142)
```

Los problemas que se nos presentaron fueron: múltiples valores en las columnas "¿A qué hora tomas el STU de CU a CU2?" y "¿A qué hora tomas el STU de CU2 a CU (Selecciona la más cercana)?" estaban separados por punto y coma, lo que dificultó las modificaciones. Además, los datos de frecuencia y satisfacción estaban en formato de texto y debieron convertirse a enteros para su manejo adecuado. También se requirió ajustar los formatos de horas, días y tiempos de espera.

Proceso de limpieza

Eliminamos las columnas con información mayormente nula que no nos es de utilidad para optimizar la base de datos y tener conclusiones más precisas y relevantes; ya que los valores no aportaran valor al análisis y puede generar confusión.

```
df=df.drop(columns=['ID', 'Hora de inicio', 'Hora de finalizaciÃ³n', 'Correo electrÃ³nico',
       'Nombre', 'Hora de la Ãºltima modificaciÃ³n','Â¿QuÃ© consideras quÃ© se debe mejorar en el STU?'])
df
[5]  ✓ 0.0s                                                    Python
```

	Tomas el STU de CU a CU2 o de CU2 a CU	GÃ©nero	Â¿CuÃ¡ntos dÃas a la semana utilizas el STU?	Â¿Tomas el STU de ida y vuelta?\n	Â¿A quÃ© hora tomas el STU de CU a CU2?	Â¿A quÃ© hora tomas el STU de CU2 a CU (Selecciona la mÃ¡s cercana)?	Â¿CuÃ¡l es tu nivel de satisfacciÃ³n con el STU?	¿CuÃ¡nto tiempo esperas el STU de CU a CU2 (minutos)?	¿CuÃ¡nto tiempo esperas el STU de CU2 a CU (minutos)?	Â¿QuÃ© tan frecuente es que viajes parado
0	No	NaN	NaN	Si	NaN	10:00;11:00;	Neutral	NaN	NaN	NaN
1	Si	MasculinoÂ	5.0	NaN	7:00;	18:00;	Satisfecho	16-30	31-45	Muy frecuente
2	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica;	17:00;14:00;15:00;16:00;18:00;	NaN	No aplica	31-45	Ocasionalmente
3	Si	MasculinoÂ	5.0	Si	7:00;	16:00;15:00;	Neutral	ene-15	31-45	bbb
4	Si	MasculinoÂ	5.0	Si	7:00;	No aplica;	Neutral	16-30	MÃ¡s de 45	Frecuente
...	...	...	...	...	...	...	...	...	...	...
3058	Si	FemeninoÂ	4.0	Si	7:00;	14:00;	Poco satisfecho	NaN	31-45	Poco frecuente
3059	NaN	FemeninoÂ	5.0	Si	10:00;13:00;	NaN	NaN	31-45	MÃ¡s de 45	Muy frecuente
3060	Si	FemeninoÂ	4.0	NaN	8:00;	14:00;	NaN	16-30	MÃ¡s de 45	Muy frecuente
3061	Si	MasculinoÂ	5.0	Si	7:00;	18:00;19:00;	NaN	ene-15	31-45	Muy frecuente
3062	Si	FemeninoÂ	5.0	Si	7:00;	19:00;	Satisfecho	16-30	MÃ¡s de 45	Ocasionalmente

Cell 6 of 36

Renombramos las columnas con caracteres inusuales o textos extensos para mejor la claridad y legibilidad de los nombres.

```
df=df.rename(columns={'Tomas el STU de CU a CU2 o de CU2 a CU':'Uso',
                      'GÃ©nero':'Genero',
                      'Â¿CuÃ¡ntos dÃas a la semana utilizas el STU?':'Dias',
                      'Â¿Tomas el STU de ida y vuelta?\n':'Ida/Vuelta',
                      'Â¿A quÃ© hora tomas el STU de CU a CU2?':'Hora Cu-Cu2',
                      'Â¿A quÃ© hora tomas el STU de CU2 a CU (Selecciona la mÃ¡s cercana)?':'Hora Cu2-Cu',
                      'Â¿CuÃ¡l es tu nivel de satisfacciÃ³n con el STU? ':'Satisfaccion',
                      '¿CuÃ¡nto tiempo esperas el STU de CU a CU2 (minutos)?':'Espera Cu-Cu2',
                      '¿CuÃ¡nto tiempo esperas el STU de CU2 a CU (minutos)?':'Espera Cu2-Cu',
                      'Â¿QuÃ© tan frecuente es que viajes parado':'Frecuencia parado'})
df
```

[6] ✓ 0.0s Python

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	No	NaN	NaN	Si	NaN	10:00;11:00;	Neutral	NaN	NaN	NaN
1	Si	MasculinoÂ	5.0	NaN	7:00;	18:00;	Satisfecho	16-30	31-45	Muy frecuente
2	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica;	17:00;14:00;15:00;16:00;18:00;	NaN	No aplica	31-45	Ocasionalmente
3	Si	MasculinoÂ	5.0	Si	7:00;	16:00;15:00;	Neutral	ene-15	31-45	bbb
4	Si	MasculinoÂ	5.0	Si	7:00;	No aplica;	Neutral	16-30	MÃ¡s de 45	Frecuente
...	...	...	...	...	...	...	...	...	...	...
3058	Si	FemeninoÂ	4.0	Si	7:00;	14:00;	Poco satisfecho	NaN	31-45	Poco frecuente
3059	NaN	FemeninoÂ	5.0	Si	10:00;13:00;	NaN	NaN	31-45	MÃ¡s de 45	Muy frecuente
3060	Si	FemeninoÂ	4.0	NaN	8:00;	14:00;	NaN	16-30	MÃ¡s de 45	Muy frecuente

Cell 6 of 36

Dividimos las columnas ‘Hora Cu-Cu2’ y ‘Hora Cu2-Cu’ que tenían múltiples valores separados por; para facilitar la manipulación de la base de datos.

```
df= df.assign(**{'Hora Cu-Cu2':df['Hora Cu-Cu2'].str.split(';')}).explode('Hora Cu-Cu2')
df=df[df['Hora Cu-Cu2'].str.strip()!='']
df=df.assign(**{'Hora Cu2-Cu':df['Hora Cu2-Cu'].str.split(';')}).explode('Hora Cu2-Cu')
df= df[df['Hora Cu2-Cu'].str.strip() != '']
df
```

[7] ✓ 0.0s Python

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	No	NaN	NaN	Si	NaN	10:00	Neutral	NaN	NaN	NaN
0	No	NaN	NaN	Si	NaN	11:00	Neutral	NaN	NaN	NaN
1	Si	MasculinoÂ	5.0	NaN	7:00	18:00	Satisfecho	16-30	31-45	Muy frecuente
2	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica	17:00	NaN	No aplica	31-45	Ocasionalmente
2	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica	14:00	NaN	No aplica	31-45	Ocasionalmente
...	...	...	...	...	...	...	...	...	...	...
3059	NaN	FemeninoÂ	5.0	Si	13:00	NaN	NaN	31-45	MÃ¡s de 45	Muy frecuente
3060	Si	FemeninoÂ	4.0	NaN	8:00	14:00	NaN	16-30	MÃ¡s de 45	Muy frecuente
3061	Si	MasculinoÂ	5.0	Si	7:00	18:00	NaN	ene-15	31-45	Muy frecuente
3061	Si	MasculinoÂ	5.0	Si	7:00	19:00	NaN	ene-15	31-45	Muy frecuente
3062	Si	FemeninoÂ	5.0	Si	7:00	19:00	Satisfecho	16-30	MÃ¡s de 45	Ocasionalmente

6339 rows × 10 columns

Cell 6 of 36

Eliminamos las filas que se encuentran duplicadas y reindexamos para evitar la redundancia de información y mantener un orden.

```
df=df.drop_duplicates()
df=df.reset_index(drop=True)
df
```

0.0s — Python

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	No	NaN	NaN	Si	NaN	10:00	Neutral	NaN	NaN	NaN
1	No	NaN	NaN	Si	NaN	11:00	Neutral	NaN	NaN	NaN
2	Si	MasculinoÂ	5.0	NaN	7:00	18:00	Satisfecho	16-30	31-45	Muy frecuente
3	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica	17:00	NaN	No aplica	31-45	Ocasionalmente
4	Si	MasculinoÂ	4.0	Solo de vuelta	No aplica	14:00	NaN	No aplica	31-45	Ocasionalmente
...	...	...	...	...	...	...	...	...	...	...
5178	Si	FemeninoÂ	4.0	Si	7:00	16:00	Poco satisfecho	16-30	31-45	Muy poco frecuente
5179	Si	FemeninoÂ	4.0	Si	7:00	14:00	Poco satisfecho	NaN	31-45	Poco frecuente
5180	NaN	FemeninoÂ	5.0	Si	10:00	NaN	NaN	31-45	MÃ¡s de 45	Muy frecuente
5181	NaN	FemeninoÂ	5.0	Si	13:00	NaN	NaN	31-45	MÃ¡s de 45	Muy frecuente
5182	Si	FemeninoÂ	4.0	NaN	8:00	14:00	NaN	16-30	MÃ¡s de 45	Muy frecuente

5183 rows × 10 columns

Pasamos los valores de espera, satisfacción y frecuencia a una escala numérica y los valores ‘bbb’ los remplazamos por el neutro para alinear los datos a la escala y estandarizar los datos, facilitando su análisis y comparación.

```
traduccion={
    'Nada satisfecho':1,
    'Poco satisfecho':2,
    'Neutral':3,
    'Satisfecho':4,
    'Totalmente satisfecho':5,
    'bbb': 3
}
df['Satisfaccion']=df['Satisfaccion'].replace(traduccion)
df['Satisfaccion'] = df['Satisfaccion'].fillna(3)
```

0.0s — Python

```
C:\Users\Danna\AppData\Local\Temp\ipykernel_8216\91697853.py:9: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in a
df['Satisfaccion']=df['Satisfaccion'].replace(traduccion)
```

```
traduccion={
    'bbb':'11:00',
    'No aplica':'11:00'
}
df[['Hora Cu-Cu2','Hora Cu2-Cu']]=df[['Hora Cu-Cu2','Hora Cu2-Cu']].replace(traduccion)
```

0.0s — Python

```
traduccion={'bbb':3}
```

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	No	NaN	NaN	Si	NaN	10:00	3.0	NaN	NaN	NaN
1	No	NaN	NaN	Si	NaN	11:00	3.0	NaN	NaN	NaN
2	Si	Masculino	5.0	NaN	7:00	18:00	4.0	30.0	45.0	Muy frecuente
3	Si	Masculino	4.0	Solo de vuelta	11:00	17:00	3.0	30.0	45.0	Ocasionalmente
4	Si	Masculino	4.0	Solo de vuelta	11:00	14:00	3.0	30.0	45.0	Ocasionalmente
...	...	...	...	...	...	...	...	...	...	...
5178	Si	Femenino	4.0	Si	7:00	16:00	2.0	30.0	45.0	Muy poco frecuente
5179	Si	Femenino	4.0	Si	7:00	14:00	2.0	NaN	45.0	Poco frecuente
5180	NaN	Femenino	5.0	Si	10:00	NaN	3.0	45.0	60.0	Muy frecuente
5181	NaN	Femenino	5.0	Si	13:00	NaN	3.0	45.0	60.0	Muy frecuente
5182	Si	Femenino	4.0	NaN	8:00	14:00	3.0	30.0	60.0	Muy frecuente

5183 rows × 10 columns

Eliminamos los valores inválidos que se encuentran en las horas para simplificar el proceso de transformación y estandarización de las horas, lo que permite asegurar que se cumpla con un formato uniforme.

```
df1=df
df1.dropna(subset=['Hora Cu-Cu2'], inplace=True)
df1.dropna(subset=['Hora Cu2-Cu'], inplace=True)
df1
```

✓ 0.0s Python

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
2	Si	Masculino	5.0	NaN	7:00	18:00	4.0	30.0	45.0	Muy frecuente
3	Si	Masculino	4.0	Solo de vuelta	11:00	17:00	3.0	30.0	45.0	Ocasionalmente
4	Si	Masculino	4.0	Solo de vuelta	11:00	14:00	3.0	30.0	45.0	Ocasionalmente
5	Si	Masculino	4.0	Solo de vuelta	11:00	15:00	3.0	30.0	45.0	Ocasionalmente
6	Si	Masculino	4.0	Solo de vuelta	11:00	16:00	3.0	30.0	45.0	Ocasionalmente
...	...	...	...	...	...	...	...	...	...	...
5173	Si	Femenino	5.0	Si	10:00	14:00	3.0	15.0	45.0	Frecuente
5177	Si	Femenino	4.0	Si	7:00	15:00	2.0	30.0	45.0	Muy poco frecuente
5178	Si	Femenino	4.0	Si	7:00	16:00	2.0	30.0	45.0	Muy poco frecuente
5179	Si	Femenino	4.0	Si	7:00	14:00	2.0	NaN	45.0	Poco frecuente
5182	Si	Femenino	4.0	NaN	8:00	14:00	3.0	30.0	60.0	Muy frecuente

4356 rows × 10 columns

Rellenamos los valores nulos de las columnas espera, frecuencia, días y satisfacción

con los promedios redondeados para evitar datos faltantes y afectar los resultados estadísticos como promedio.

```
promediocucu2= df1['Espera Cu-Cu2'].mean()
promediocucu2=round(promediocucu2)
promediocu2cu= df1['Espera Cu2-Cu'].mean()
promediocu2cu=round(promediocu2cu)
promediosat=df1['Satisfaccion'].mean()
promediosat=round(promediosat)
promediofec=df1['Frecuencia parado'].mean()
promediofec=round(promediofec)
promediodia=df1['Dias'].mean()
promediodia=round(promediodia)
print(promediocu2cu , promediocucu2 , promediodia , promediofec , promediosat)
```

[16] Python

```
44 29 5 4 2
```

```
df1['Espera Cu-Cu2'].fillna(44, inplace=True)
df1['Espera Cu2-Cu'].fillna(29, inplace=True)
df1['Satisfaccion'].fillna(2,inplace=True)
df1['Frecuencia parado'].fillna(4,inplace=True)
df1['Dias'].fillna(5,inplace=True)
df
```

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
2	Si	Masculino	5.0	NaN	1900-01-01 07:00:00	1900-01-01 18:00:00	4.0	30.0	45.0	5.0
3	Si	Masculino	4.0	Solo de vuelta	1900-01-01 11:00:00	1900-01-01 17:00:00	3.0	30.0	45.0	3.0
4	Si	Masculino	4.0	Solo de vuelta	1900-01-01 11:00:00	1900-01-01 14:00:00	3.0	30.0	45.0	3.0
5	Si	Masculino	4.0	Solo de vuelta	1900-01-01 11:00:00	1900-01-01 15:00:00	3.0	30.0	45.0	3.0
6	Si	Masculino	4.0	Solo de vuelta	1900-01-01 11:00:00	1900-01-01 16:00:00	3.0	30.0	45.0	3.0
...	...	...	...	...	...	...	...	...	...	...
5173	Si	Femenino	5.0	Si	1900-01-01 10:00:00	1900-01-01 14:00:00	3.0	15.0	45.0	4.0
5177	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	2.0	30.0	45.0	1.0
5178	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 16:00:00	2.0	30.0	45.0	1.0
5179	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 14:00:00	2.0	NaN	45.0	2.0
5182	Si	Femenino	4.0	NaN	1900-01-01 08:00:00	1900-01-01 14:00:00	3.0	30.0	60.0	5.0

Rellenamos los valores nulos de la columna genero por S/G y reindexamos con el fin de que se contara con información completa para el análisis.

```
df1=df1.reset_index(drop=True)
df1
```

Python

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 16:00:00	3.0	15.0	45.0	3.0
1	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	3.0	15.0	45.0	3.0
2	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 11:00:00	3.0	30.0	60.0	4.0
3	Si	Masculino	3.0	Si	1900-01-01 08:00:00	1900-01-01 18:00:00	3.0	30.0	45.0	5.0
4	Si	Femenino	5.0	Si	1900-01-01 11:00:00	1900-01-01 18:00:00	2.0	15.0	45.0	5.0
...	...	...	...	...	...	...	...	...	...	...
3176	Si	Femenino	5.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	2.0	15.0	60.0	5.0
3177	Si	Femenino	5.0	Si	1900-01-01 10:00:00	1900-01-01 14:00:00	3.0	15.0	45.0	4.0
3178	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	2.0	30.0	45.0	1.0
3179	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 16:00:00	2.0	30.0	45.0	1.0
3180	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 14:00:00	2.0	44.0	45.0	2.0

3181 rows × 10 columns

	Uso	Genero	Dias	Ida/Vuelta	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
0	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 16:00:00	3.0	15.0	45.0	3.0
1	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	3.0	15.0	45.0	3.0
2	Si	Masculino	5.0	Si	1900-01-01 07:00:00	1900-01-01 11:00:00	3.0	30.0	60.0	4.0
3	Si	Masculino	3.0	Si	1900-01-01 08:00:00	1900-01-01 18:00:00	3.0	30.0	45.0	5.0
4	Si	Femenino	5.0	Si	1900-01-01 11:00:00	1900-01-01 18:00:00	2.0	15.0	45.0	5.0
...	...	...	...	...	...	...	...	...	...	...
3176	Si	Femenino	5.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	2.0	15.0	60.0	5.0
3177	Si	Femenino	5.0	Si	1900-01-01 10:00:00	1900-01-01 14:00:00	3.0	15.0	45.0	4.0
3178	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 15:00:00	2.0	30.0	45.0	1.0
3179	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 16:00:00	2.0	30.0	45.0	1.0
3180	Si	Femenino	4.0	Si	1900-01-01 07:00:00	1900-01-01 14:00:00	2.0	44.0	45.0	2.0

Finalmente convertimos los tipos de valores para estandarizar los datos de cada columna y sean manejables de manera más eficiente

```
df1['Satisfaccion'] = df1['Satisfaccion'].astype(int)
df1['Espera Cu-Cu2'] = df1['Espera Cu-Cu2'].astype(int)
df1['Espera Cu2-Cu'] = df1['Espera Cu2-Cu'].astype(int)
df1['Frecuencia parado'] = df1['Frecuencia parado'].astype(int)
df1['Dias']=df1['Dias'].astype(int)
```

Python

Resultados

Descripción estadística de la base de datos después de realizar la limpieza

	Dias	Hora Cu-Cu2	Hora Cu2-Cu	Satisfaccion	Espera Cu-Cu2	Espera Cu2-Cu	Frecuencia parado
count	3181.000000	3181	3181	3181.000000	3181.000000	3181.000000	3181.000000
mean	4.770512	1900-01-01 08:59:24.916692736	1900-01-01 15:19:17.749135104	2.457718	29.967935	43.467149	3.997799
min	1.000000	1900-01-01 07:00:00	1900-01-01 10:00:00	1.000000	15.000000	15.000000	1.000000
25%	5.000000	1900-01-01 07:00:00	1900-01-01 14:00:00	2.000000	15.000000	30.000000	3.000000
50%	5.000000	1900-01-01 09:00:00	1900-01-01 15:00:00	3.000000	30.000000	45.000000	4.000000
75%	5.000000	1900-01-01 10:00:00	1900-01-01 17:00:00	3.000000	44.000000	60.000000	5.000000
max	5.000000	1900-01-01 15:00:00	1900-01-01 19:00:00	5.000000	60.000000	60.000000	5.000000
std	0.595706	NaN	NaN	0.839205	14.322199	13.333669	0.910972

Tabla en donde se muestra el porcentaje de valore faltantes y el tipo de dato por columna.

	Porcentaje de Valores Faltantes	Tipo de Dato
Uso	0.0	object
Genero	0.0	object
Dias	0.0	int64
Ida/Vuelta	0.0	object
Hora Cu-Cu2	0.0	datetime64[ns]
Hora Cu2-Cu	0.0	datetime64[ns]
Satisfaccion	0.0	int64
Espera Cu-Cu2	0.0	int64
Espera Cu2-Cu	0.0	int64
Frecuencia parado	0.0	int64