In [ ]:
```
Gerardo Carcoba - A01178753 - Carrera:LAF
```

Comprensión de los Datos

In [11]:
```
#importa librerías
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

Descripción de Variables

Pregnancies: Cuántas veces ha estado embarazada la persona
Glucose: Nivel de azúcar en la sangre después de una prueba
BloodPressure: Presión arterial medida en reposo
SkinThickness: Grosor de la piel del brazo (mide grasa corporal)
Insulin: Cantidad de insulina en la sangre
BMI: Índice de masa corporal (relación entre peso y altura)
DiabetesPedigreeFunction: Qué tan probable es tener diabetes según los antecedentes familiares
Age: Edad de la persona
Outcome: Resultado del examen (0 = No tiene diabetes, 1 = Sí tiene diabetes)

Ejemplo: Crear un objeto DataFrame con base en un archivo .csv

In [12]:
```
#lee archivo csv
df = pd.read_csv("diabetes.csv")
```

In [13]:
```
#Usa función shape para revisar el total de renglones y columnas
df.shape
```

Out[13]:
```
(768, 9)
```

In [14]:
```
#Revisa los primeros 5 renglones del dataset usando la función head()
df.head ()
```

Out[14]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	

In [56]:
```
#Revisa los últimos 5 renglones del dataset usando la función tail()
df.tail()
```

Out[56]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

In [57]:
```
#Revisa la información mas completa del conjunto de datos usando la función
#Muestra el total de datos, las columnas y su tipo correspondiente, dice si
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

In [54]:
```
#revisa cuántos valores únicos tiene cada atributo del archivo usando la fur
df.nunique()
```

Out[54]:
```
Pregnancies                 17
Glucose                    136
BloodPressure               47
SkinThickness               51
Insulin                    186
BMI                        248
DiabetesPedigreeFunction   517
Age                         52
Outcome                      2
dtype: int64
```

Exploración de Datos

In [26]:
```
#utiliza la función describe() para obtener estadística básica. se puede inc
df.describe()
```

Out[26]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	B
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.0000
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.9925
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.8841
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.0000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.3000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.0000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.6000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.1000

In [ ]:
```
Los datos muestran una glucosa promedio de 120.9 entre los participantes y u
```

In [65]:
```
df["BMI"].describe()
```

Out[65]:
```
count    768.000000
mean      31.992578
std        7.884160
min        0.000000
25%       27.300000
50%       32.000000
75%       36.600000
max       67.100000
Name: BMI, dtype: float64
```

In [ ]:
```
La mayoría de los participantes tienen sobrepeso o obesidad, lo que aumenta
```

In [66]:
```
df["Outcome"].describe()
```

Out[66]:
```
count    768.000000
mean       0.348958
std        0.476951
min        0.000000
25%        0.000000
50%        0.000000
75%        1.000000
max        1.000000
Name: Outcome, dtype: float64
```

In [ ]:
```
Esta grafica muestra que hay un total de 768 registros de participantes y el
```

In [30]:
```
#Revisa Valores nulos con funcion isnull().sum()
df.isnull().sum()
```

Out[30]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [ ]:
```
Esta grafica demuestra que no hay en ninguna variable valores nulo, signific
```

In [62]:
```
#Revisar valores únicos por columna usando función unique(): nombre-columna.
df["Glucose"].unique()
```

Out[62]:
```
array([148,  85, 183,  89, 137, 116,  78, 115, 197, 125, 110, 168, 139,
       189, 166, 100, 118, 107, 103, 126,  99, 196, 119, 143, 147,  97,
       145, 117, 109, 158,  88,  92, 122, 138, 102,  90, 111, 180, 133,
       106, 171, 159, 146,  71, 105, 101, 176, 150,  73, 187,  84,  44,
       141, 114,  95, 129,  79,   0,  62, 131, 112, 113,  74,  83, 136,
        80, 123,  81, 134, 142, 144,  93, 163, 151,  96, 155,  76, 160,
       124, 162, 132, 120, 173, 170, 128, 108, 154,  57, 156, 153, 188,
       152, 104,  87,  75, 179, 130, 194, 181, 135, 184, 140, 177, 164,
        91, 165,  86, 193, 191, 161, 167,  77, 182, 157, 178,  61,  98,
       127,  82,  72, 172,  94, 175, 195,  68, 186, 198, 121,  67, 174,
       199,  56, 169, 149,  65, 190])
```

In [ ]:
```
La variable glucosa tiene gran variedad de valores, indicando que hay difere
```

In [67]:
```
#Revisar valores únicos por columna usando función unique(): nombre-columna.
df["BMI"].unique()
```

Out[67]:
```
array([33.6, 26.6, 23.3, 28.1, 43.1, 25.6, 31. , 35.3, 30.5,  0. , 37.6,
       38. , 27.1, 30.1, 25.8, 30. , 45.8, 29.6, 43.3, 34.6, 39.3, 35.4,
       39.8, 29. , 36.6, 31.1, 39.4, 23.2, 22.2, 34.1, 36. , 31.6, 24.8,
       19.9, 27.6, 24. , 33.2, 32.9, 38.2, 37.1, 34. , 40.2, 22.7, 45.4,
       27.4, 42. , 29.7, 28. , 39.1, 19.4, 24.2, 24.4, 33.7, 34.7, 23. ,
       37.7, 46.8, 40.5, 41.5, 25. , 25.4, 32.8, 32.5, 42.7, 19.6, 28.9,
       28.6, 43.4, 35.1, 32. , 24.7, 32.6, 43.2, 22.4, 29.3, 24.6, 48.8,
       32.4, 38.5, 26.5, 19.1, 46.7, 23.8, 33.9, 20.4, 28.7, 49.7, 39. ,
       26.1, 22.5, 39.6, 29.5, 34.3, 37.4, 33.3, 31.2, 28.2, 53.2, 34.2,
       26.8, 55. , 42.9, 34.5, 27.9, 38.3, 21.1, 33.8, 30.8, 36.9, 39.5,
       27.3, 21.9, 40.6, 47.9, 50. , 25.2, 40.9, 37.2, 44.2, 29.9, 31.9,
       28.4, 43.5, 32.7, 67.1, 45. , 34.9, 27.7, 35.9, 22.6, 33.1, 30.4,
       52.3, 24.3, 22.9, 34.8, 30.9, 40.1, 23.9, 37.5, 35.5, 42.8, 42.6,
       41.8, 35.8, 37.8, 28.8, 23.6, 35.7, 36.7, 45.2, 44. , 46.2, 35. ,
       43.6, 44.1, 18.4, 29.2, 25.9, 32.1, 36.3, 40. , 25.1, 27.5, 45.6,
       27.8, 24.9, 25.3, 37.9, 27. , 26. , 38.7, 20.8, 36.1, 30.7, 32.3,
       52.9, 21. , 39.7, 25.5, 26.2, 19.3, 38.1, 23.5, 45.5, 23.1, 39.9,
       36.8, 21.8, 41. , 42.2, 34.4, 27.2, 36.5, 29.8, 39.2, 38.4, 36.2,
       48.3, 20. , 22.3, 45.7, 23.7, 22.1, 42.1, 42.4, 18.2, 26.4, 45.3,
       37. , 24.5, 32.2, 59.4, 21.2, 26.7, 30.2, 46.1, 41.3, 38.8, 35.2,
       42.3, 40.7, 46.5, 33.5, 37.3, 30.3, 26.3, 21.7, 36.4, 28.5, 26.9,
       38.6, 31.3, 19.5, 20.1, 40.8, 23.4, 28.3, 38.9, 57.3, 35.6, 49.6,
       44.6, 24.1, 44.5, 41.2, 49.3, 46.3])
```

In [ ]:
```
Hay una gran variedad entre los valores del BMI, indicando que hay participa
```

In [131…
```
#Revisar valores únicos por columna usando función unique(): nombre-columna.
df["Outcome"].unique()
```

Out[131…

```
array([181.6, 111.6, 206.3, 117.1, 180.1, 141.6, 109. , 150.3, 227.5,
       125. , 147.6, 206. , 166.1, 219.1, 191.8, 130. , 163.8, 136.6,
       146.3, 149.6, 165.3, 134.4, 235.8, 148. , 179.6, 156.1, 186.4,
       120.2, 167.2, 151.1, 145. , 189.6, 112.8, 111.9, 127. , 171.2,
       134.9, 128.2, 148.1, 214. , 173.2, 128.7, 216.4, 222. , 175.7,
        99. , 142.1, 105. , 122.4, 125.2, 112.4, 209.7, 184.7,  96. ,
       224.7, 146.8, 186.5, 146.5,  84. , 165.9,  69. , 166.4, 128. ,
       141.5, 151.7, 114.6, 174.9, 132.9, 167.6, 169.4, 164.1, 111. ,
        24.7,  94.6, 132.7, 174.2, 137. , 135.4,  74. , 112.3, 125.6,
       185.8, 142.4, 142.6, 138.5, 173.1, 133.5,  99.1, 155. , 127.7,
       157.8, 166.7, 177.9, 123.6,  91.4, 121.7, 171.7, 202. , 177.1,
       147.5, 107.6, 124.6, 154.7, 118.4, 173.5, 117.3, 132.4, 204.3,
       189. , 110. , 190.5, 177.2, 158. , 111.7, 122.2, 215.2, 145.2,
       140.6, 158.8, 143. , 162.9, 151.3, 151.5, 202.7, 155.3, 204.5,
       122.3, 130.8, 160.2, 141.9, 149.1, 145.5, 140.5, 140.4, 186.8,
       102. ,  89.8, 136.5, 180.7, 173.4, 135.9, 190.3, 193.6, 235.9,
       134.2, 117. , 203.9, 139.2, 158.2, 129.7, 162.6, 133.9, 180.5,
       139.4, 132.8, 115.9, 122.5, 104.7, 211.7, 116.2, 196.1, 188. ,
       169.1, 110.2, 153.9,  27.7,  99.8, 168.6, 229.9, 211.1, 160. ,
       136.9, 170.6, 133.6, 189.4, 187.3, 109.4, 197.4, 129.3, 129.9,
       143.8, 178.9, 144. , 178.1, 135.3, 119.4, 140.7, 233.5, 199.7,
       129.2, 219.5, 108.7, 189.8, 213.2, 182.6, 146.2, 192.8, 144.8,
       114. , 149.8, 211.6, 144.2, 170.8, 121.6, 136.7, 199.2, 233.7,
       162.2, 186. , 180.2, 104.4, 157. , 103.7, 214.6, 216.9, 223.1,
       194.8, 124.1, 164.6, 146.1, 184.2, 153.2, 217.3, 159.4, 141.1,
       137.2, 114.4, 121.8, 119.6, 146.6, 142.7, 218.9, 188.3, 221.9,
       171. , 127.1, 174.4, 129.6, 174.3, 168. , 133.2, 104.2, 140.2,
       132.5, 131.8, 138.9, 133.3, 183.9, 164.9, 165.4, 191.4, 135. ,
       162. , 193.7, 116.8, 144.1, 114.9, 143.6, 171.3, 168.5, 182.9,
       174. , 156.7, 135.6, 199.3, 175.6, 167.9, 159.7, 161.8, 158.5,
       156.9, 106.2, 145.4, 142.5, 144.9, 118.3, 212.5, 153.1, 217.5,
       156.5, 151.4, 178.8, 147.7, 157.1, 214.1, 135.8, 119.7, 223.3,
       118.9, 212.9, 150.8, 186.2, 217.9, 155.9, 120.9,  32. , 164.5,
       167.7, 139.5, 120.8,  41. , 168.2,  95.4, 117.2, 195.4, 158.3,
       168.9, 123.3, 232.5, 220.2, 187.8, 142.2, 184.5, 181.9, 133. ,
       151.6, 122. , 108.5, 165.8, 211.4, 118. , 119.8, 139.9, 179.2,
       123.2, 124.2, 136.4, 137.8, 115.1, 149.3, 141.3, 148.3, 176. ,
       131.6, 132. , 138.1, 190.9, 120.7, 103.1, 227.9, 161.2, 103.6,
       200.9, 165.1, 143.9, 122.9, 222.9, 214.4, 137.7, 146.4, 185.4,
       169.2, 172.6, 208.7, 182.5, 101.2, 155.4, 164.3, 120. , 145.8,
       193.9, 221. ,  94. , 215.1, 121.2, 114.5, 183.4, 177.4, 176.9,
       115.2, 150.2, 146.7, 239.4, 125.3, 131.5, 137.6, 150.5, 103.2,
       130.9, 138.6, 208.6, 161.7, 185.6, 159.9, 140.8,  92.8, 109.3,
       115.6, 139. , 101.8, 133.8, 150. , 200.1, 185.3, 170.2, 142.9,
       138.7, 151.9, 193.5, 189.2, 177.3, 179.7, 220.1,  80. , 192.6,
       136. , 111.1, 183.3,  39. , 127.3, 108.3, 216.5, 158.6, 113.7,
       161.1, 194.6, 175.3, 108.8, 148.6,  93. , 157.2, 164.2, 156.6,
       142.3, 147.8, 151.8, 152.3, 110.3, 134.6,  78.7, 163.3, 160.4,
       124.9, 123.5, 220.5, 230.6, 196.8, 143.4, 141.8, 117.9, 149.5,
       135.1, 149.2, 239.3, 125.9, 123.4, 121.1, 124. , 110.5, 152.6,
       154.5, 160.9, 231.7, 193.1, 134. , 152.7, 115.5, 209.3,  94.1,
       220. , 209.8, 132.1, 185.2, 159.8, 111.5, 206.9, 206.2, 137.5,
       174.1, 145.7,  88.1, 150.4, 223.8, 138.8, 149.7, 164.4, 118.7,
       141.4, 155.5, 135.2, 137.9, 119.5, 131.3, 162.3, 176.5, 130.6,
       196.4, 190.4, 216.8, 133.4, 122.8, 158.9, 166. , 114.2, 189.7,
       241.9, 204.6, 148.7, 177.5, 184.9, 198.6, 124.4, 103.5, 180.3,
```

```
       126.6, 225.9, 180.8, 128.3,  80.2, 139.6, 157.3, 190.1, 200.3,
       160.1, 167.5, 113.5, 172.4, 198.9, 161.5, 162.5, 157.9, 203. ,
       138.4, 115. , 161.4, 131. , 155.6, 220.9, 206.8, 143.5, 132.6,
       152.1, 178.3, 197.9, 122.1, 158.4, 148.4, 218.5, 130.4, 153.4,
        97. , 172.7, 196.3, 223.4, 186.3, 224.3, 169. , 159.3, 225.5,
       116.4, 147.2])
```

In [ ]:
```
La variable Outcome muestra tiene gran variedad en sus valores, indicando di
```

In [68]:
```
df["Outcome"].unique()
```

Out[68]:
```
array([1, 0])
```

Variables Cuantitativas

Medidas de tendencia central

In [43]:
```
#Glucose
#Se puede obtener la media, mediana y moda para
mean_Glucose = df['Glucose'].mean()
median_Glucose =df['Glucose'].median()
mode_Glucose = df['Glucose'].mode()
print("Mean_Glucose:",mean_Glucose)
print("Median_Glucose:",median_Glucose)
print("Mode_Glucose:",mode_Glucose)
```

```
Mean_Glucose: 120.89453125
Median_Glucose: 117.0
Mode_Glucose: 0     99
1    100
Name: Glucose, dtype: int64
```

Conclusiones:

La Glucose promedio fue 121

La Glucose al centro es 117 La Glucose más repetida fue de 99

In [133…
```
#BMI
#Se puede obtener la media, mediana y moda para
mean_BMI = df['BMI'].mean()
median_BMI =df['BMI'].median()
mode_BMI = df['BMI'].mode()
print("Mean_BMI:",mean_BMI)
print("Median_BMI:",median_BMI)
print("Mode_BMI:",mode_BMI)
```

```
Mean_BMI: 31.992578124999998
Median_BMI: 32.0
Mode_BMI: 0    32.0
Name: BMI, dtype: float64
```

Conclusiones:

La BMI promedio fue 32

La BMI al centro es 32

La BMI más repetida fue de 32

In [132…
```
#Outcome
#Se puede obtener la media, mediana y moda para
mean_Outcome = df['Outcome'].mean()
median_Outcome =df['Outcome'].median()
mode_Outcome = df['Outcome'].mode()
print("Mean_Outcome:",mean_Outcome)
print("Median_Outcome:",median_Outcome)
print("Mode_Outcome:",mode_Outcome)
```

```
Mean_Outcome: 152.887109375
Median_Outcome: 148.3
Mode_Outcome: 0     115.2
1     116.2
2     124.6
3     127.3
4     131.8
5     137.8
6     146.8
7     151.4
8     156.1
9     158.8
10    160.0
Name: Outcome, dtype: float64
```

Conclusiones:

El Outcome promedio fue 153 El Outcome al centro es 148 El Outcome más repetida fue de 115

Variables Categóricas

In [72]:
```
#Para conteo  de cada valor en una columna, en orden descendente usar funció
# nombreDataframe.columna.value_counts()
# nombreDataframe['columna'].value_counts()
df.Glucose.value_counts()
```

Out[72]:
```
Glucose
99     17
100    17
111    14
125    14
129    14
       ..
56      1
169     1
149     1
65      1
190     1
Name: count, Length: 136, dtype: int64
```

In [ ]:
```
La tabla muestra que los valores de glucosa más comunes estan entre 99 y 100
```

In [108…
```
#Para conteo  de cada valor en una columna, en orden descendente usar funció
# nombreDataframe.columna.value_counts()
# nombreDataframe['columna'].value_counts()
df.BMI.value_counts()
```

Out[108…
```
BMI
32.0    13
31.6    12
31.2    12
0.0     11
32.4    10
        ..
49.6     1
24.1     1
41.2     1
49.3     1
46.3     1
Name: count, Length: 248, dtype: int64
```

In [ ]:
```
Los valor mmas comunes son 31 y 32 indicando que la mayoria de los pacientes
```

In [109…
```
#Para conteo  de cada valor en una columna, en orden descendente usar funció
# nombreDataframe.columna.value_counts()
# nombreDataframe['columna'].value_counts()
df.Outcome.value_counts()
```

Out[109…
```
Outcome
156.1    4
131.8    4
124.6    4
151.4    4
137.8    4
        ..
223.4    1
196.3    1
172.7    1
97.0     1
153.4    1
Name: count, Length: 551, dtype: int64
```

In [ ]:
```
Los valores más frecuentes del Outcome se concentran entre 124 y 156, demost
```

In [34]:
```
#Revisa conteo de varias columnas
columnas = ['Glucose', 'BMI', 'Outcome']
for col in columnas:
    print(f"\nConteo de valores únicos en la columna: {col}")
    print(df[col].value_counts())
```

```
Conteo de valores únicos en la columna: Glucose
Glucose
99     17
100    17
111    14
125    14
129    14
       ..
56      1
169     1
149     1
65      1
190     1
Name: count, Length: 136, dtype: int64

Conteo de valores únicos en la columna: BMI
BMI
32.0    13
31.6    12
31.2    12
0.0     11
32.4    10
        ..
49.6     1
24.1     1
41.2     1
49.3     1
46.3     1
Name: count, Length: 248, dtype: int64

Conteo de valores únicos en la columna: Outcome
Outcome
0    500
1    268
Name: count, dtype: int64
```

In [ ]:
```
# Crear variable totalScore que incluya la suma de las columnas Outcome, Glu
# Mostrar los registros donde el total sea mayor o igual a 18
df["Outcome"] + df["Glucose"] + df["BMI"] >= 18
```

In [97]:
```
df
```

Out[97]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

In [ ]:
```
En la grafica podemos ver qué pacientes tienen un nivel combinado alto entre
```

Consulta

In [112…
```
# df.iloc[i]: Accede a la fila en la posición i.
# Acceder a la primera fila
df.iloc[0]
```

Out[112…
```
Pregnancies                 6.000
Glucose                   148.000
BloodPressure              72.000
SkinThickness              35.000
Insulin                     0.000
BMI                        33.600
DiabetesPedigreeFunction    0.627
Age                        50.000
Outcome                   181.600
Name: 0, dtype: float64
```

In [ ]:
```
El primer registro de los personas muestra valores elevados de glucosa ubica
```

In [99]:
```
# Acceder a las dos primeras filas
df.iloc[:2]
```

Out[99]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	

In [100…
```
#Seleccionar columnas, indicando entre corchetes [nombreColumna, nombreColum
df[["Glucose", "BMI", "Outcome"]]
```

Out[100…

	Glucose	BMI	Outcome
0	148	33.6	181.6
1	85	26.6	111.6
2	183	23.3	206.3
3	89	28.1	117.1
4	137	43.1	180.1
...	...	...	...
763	101	32.9	133.9
764	122	36.8	158.8
765	121	26.2	147.2
766	126	30.1	156.1
767	93	30.4	123.4

768 rows × 3 columns

In [ ]:
```
Las columnas seleccionadas muestran que los valores de Glucose, BMI y Outcom
```

In [114…
```
#Selección de filas [indicar dataframe[columna] operador valor]
df[df["Glucose"] >= 150]
```

Out[114…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
2	8	183	64	0	0	23.3	
8	2	197	70	45	543	30.5	
11	10	168	74	0	0	38.0	
13	1	189	60	23	846	30.1	
14	5	166	72	19	175	25.8	
...	...	...	...	...	...	...	
749	6	162	62	0	0	24.3	
753	0	181	88	44	510	43.3	
754	8	154	78	32	0	32.4	
759	6	190	92	0	0	35.5	
761	9	170	74	31	0	44.0	

143 rows × 9 columns

In [ ]:
```
Los datos seleccionados muestran que 143 registros presentan niveles de gluc
```

In [115…
```
#Selección de filas [indicar dataframe[columna] operador valor]
df[df["Outcome"] >= 150]
```

Out[115…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
0	6	148	72	35	0	33.6	
2	8	183	64	0	0	23.3	
4	0	137	40	35	168	43.1	
7	10	115	0	0	0	35.3	
8	2	197	70	45	543	30.5	
…	…	…	…	…	…	…	
757	0	123	72	0	0	36.3	
759	6	190	92	0	0	35.5	
761	9	170	74	31	0	44.0	
764	2	122	70	27	0	36.8	
766	1	126	60	0	0	30.1	

371 rows × 9 columns

In [ ]:
```
Se identificaron 371 registros con valores de Outcome mayores o iguales a 15
```

In [121…
```
#Selección de filas [indicar dataframe[columna] operador valor]
df[df["BMI"] > 50]
```

Out[121…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
120	0	162	76	56	100	53.2	
125	1	88	30	42	99	55.0	
177	0	129	110	46	130	67.1	
193	11	135	0	0	0	52.3	
247	0	165	90	33	680	52.3	
303	5	115	98	0	0	52.9	
445	0	180	78	63	14	59.4	
673	3	123	100	35	240	57.3	

In [ ]:
```
Los registros con BMI mayor a 50 esta muy relacionado a las personas que tie
```

In [104…
```
#ordenar usando funcion sort_values(by=atributo, ascending=True/false)
```

```
df.sort_values(by="BMI", ascending=True)
```

Out[104…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
49	7	105	0	0	0	0.0	
60	2	84	0	0	0	0.0	
706	10	115	0	0	0	0.0	
81	2	74	0	0	0	0.0	
684	5	136	82	0	0	0.0	
...	...	...	...	...	...	...	
120	0	162	76	56	100	53.2	
125	1	88	30	42	99	55.0	
673	3	123	100	35	240	57.3	
445	0	180	78	63	14	59.4	
177	0	129	110	46	130	67.1	

768 rows × 9 columns

In [122…

```
#ordenar usando funcion sort_values(by=atributo, ascending=True/false)
df.sort_values(by="Glucose", ascending=True)
```

Out[122…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
75	1	0	48	20	0	24.7	
349	5	0	80	32	0	41.0	
342	1	0	68	35	0	32.0	
502	6	0	68	41	0	39.0	
182	1	0	74	20	23	27.7	
...	...	...	...	...	...	...	
408	8	197	74	0	0	25.9	
579	2	197	70	99	0	34.7	
228	4	197	70	39	744	36.7	
561	0	198	66	32	274	41.3	
661	1	199	76	43	0	42.9	

768 rows × 9 columns

In [123…

```
#ordenar usando funcion sort_values(by=atributo, ascending=True/false)
df.sort_values(by="Outcome", ascending=True)
```

Out[123…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
75	1	0	48	20	0	24.7	
182	1	0	74	20	23	27.7	
342	1	0	68	35	0	32.0	
502	6	0	68	41	0	39.0	
349	5	0	80	32	0	41.0	
…	…	…	…	…	…	…	
22	7	196	90	0	0	39.8	
154	8	188	78	0	0	47.9	
561	0	198	66	32	274	41.3	
445	0	180	78	63	14	59.4	
661	1	199	76	43	0	42.9	

768 rows × 9 columns

In [105…

```
#Agrupar por un atributo y calcular función de agregación utilizando groupby
df.groupby("Outcome")[["Glucose", "BMI"]].mean()
```

Out[105…

Outcome	Glucose	BMI
24.7	0.0	24.7
27.7	0.0	27.7
32.0	0.0	32.0
39.0	0.0	39.0
41.0	0.0	41.0
…	…	…
235.8	196.0	39.8
235.9	188.0	47.9
239.3	198.0	41.3
239.4	180.0	59.4
241.9	199.0	42.9

551 rows × 2 columns

Crea un subconjunto de diabetes para paciente con glucos mayor a 100

In [35]:
```
# Usa el criterio para extraer solo los pacientes con glucosa alta (Glucose
glucosa_alta = df[df["Glucose"] > 100]
```

In [36]:
```
df
```

Out[36]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

In [ ]:
```
En la tabla hay un filtro que filtra a los participantes con niveles altos d
```

Crea un subconjunto de diabetes para paciente con BMI Menor a 30

In [127…
```
# Usa el criterio para extraer solo los pacientes con BMI baja (BMI > 30)
glucosa_alta = df[df["BMI"] > 30]
```

In [129…
```
df
```

Out[129…

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
…	…	…	…	…	…	…	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

In [ ]:
```
En la tabla se filtro para solo mostrar las personas que tiene sobre peso ya
```

Crea un subconjunto de diabetes para el resultado con Outcome mayor a 0

In [40]:
```
# Usa el criterio para extraer solo los resultado con 140 o mayor (Outcome >
glucosa_alta = df[df["Outcome"] > 0]
```

In [39]:
```
df
```

Out[39]:

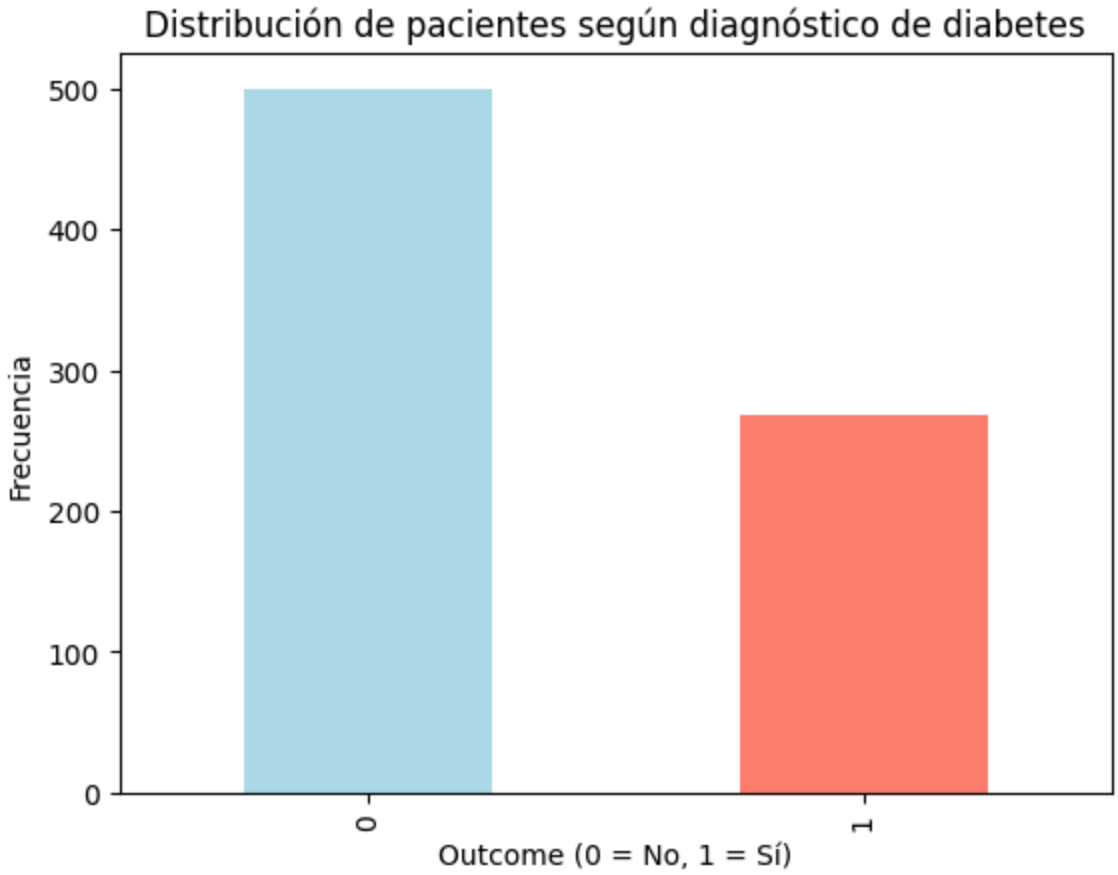
	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedi
0	6	148	72	35	0	33.6	
1	1	85	66	29	0	26.6	
2	8	183	64	0	0	23.3	
3	1	89	66	23	94	28.1	
4	0	137	40	35	168	43.1	
...	...	...	...	...	...	...	
763	10	101	76	48	180	32.9	
764	2	122	70	27	0	36.8	
765	5	121	72	23	112	26.2	
766	1	126	60	0	0	30.1	
767	1	93	70	31	0	30.4	

768 rows × 9 columns

In [ ]:
```
La grafica filtra a toda las personas que tiene un resultado en el outcome c
```

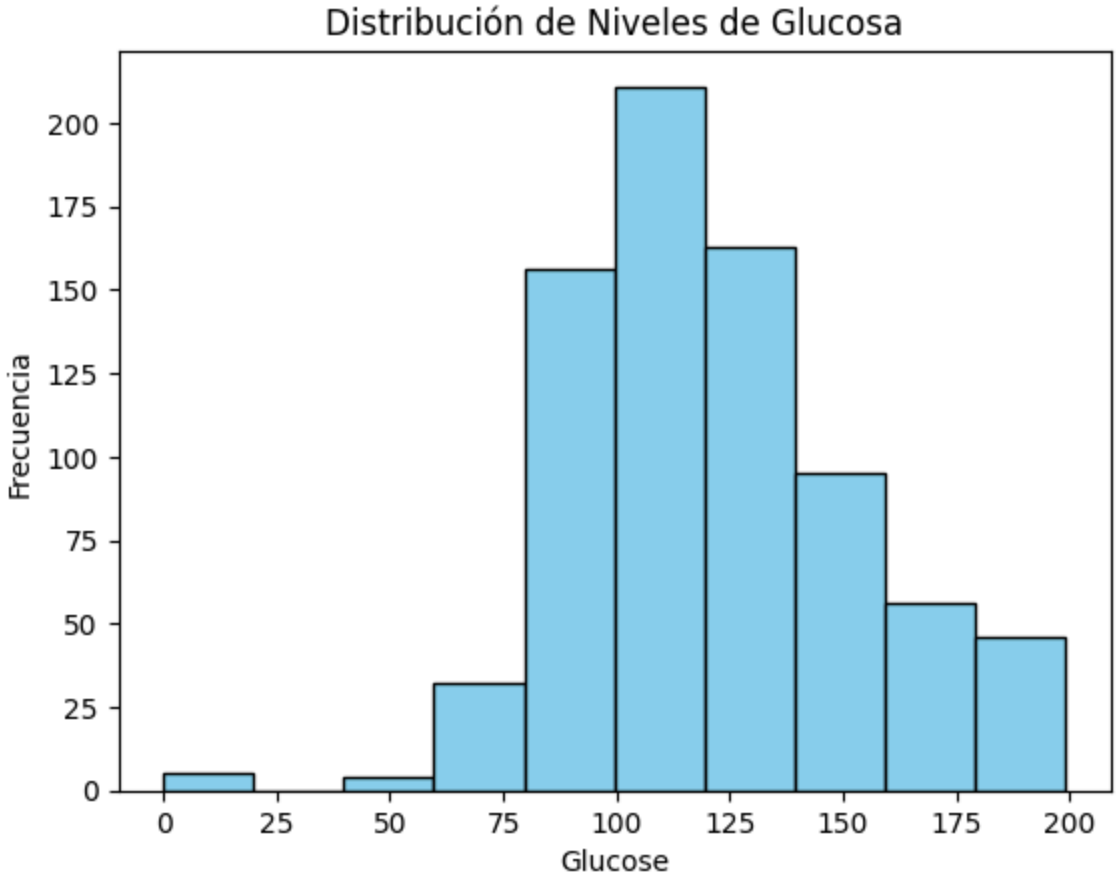
Consulta

In [17]:
```
# Distribución de Outcome
outcome = df['Outcome'].value_counts()
outcome.plot(kind='bar', color=['lightblue', 'salmon'])
plt.title('Distribución de pacientes según diagnóstico de diabetes')
plt.xlabel('Outcome (0 = No, 1 = Sí)')
plt.ylabel('Frecuencia')
plt.show()
```



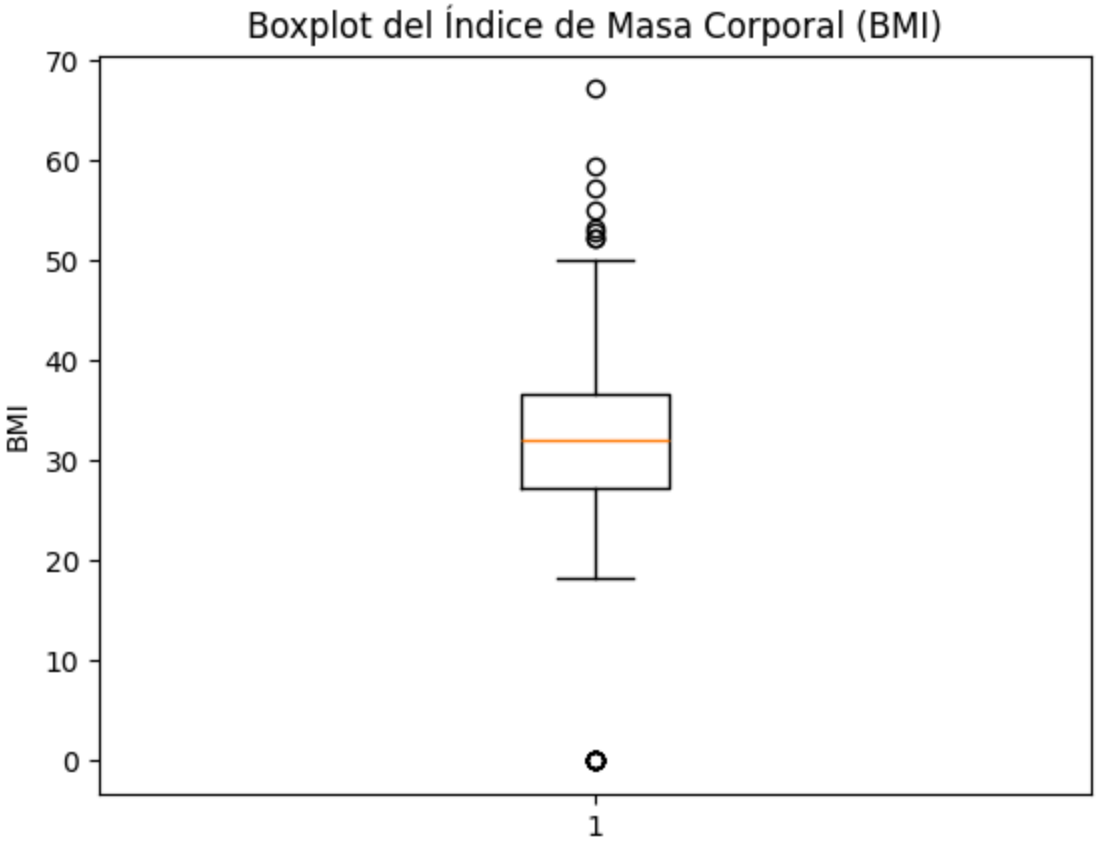
In [ ]:
```
La gráfica muestra que la mayoría de los participantes no tienen diabetes, m
```

In [18]:
```
# Histograma de Glucose
plt.hist(df['Glucose'], bins=10, color='skyblue', edgecolor='black')
plt.title('Distribución de Niveles de Glucosa')
plt.xlabel('Glucose')
plt.ylabel('Frecuencia')
plt.show()
```



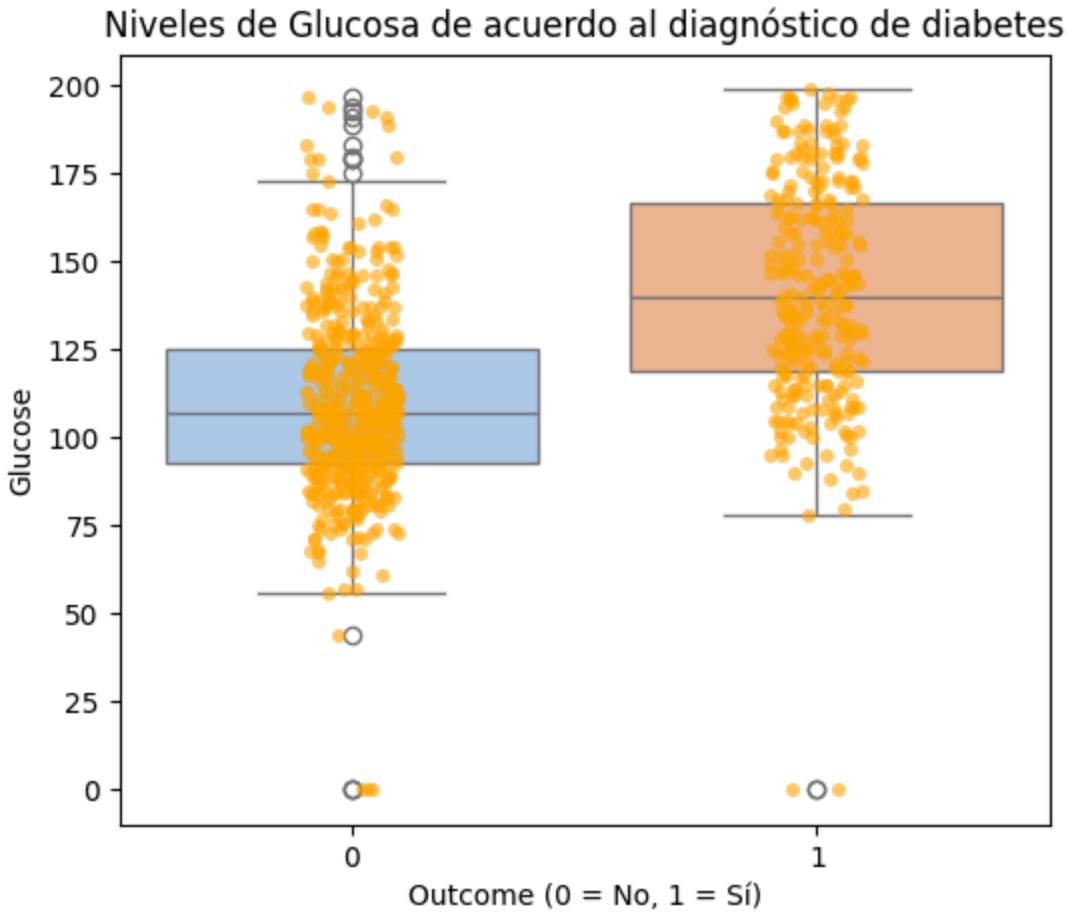
In [ ]:
```
La gráfica muestra que la mayoría de los pacientes tienen el nivel de la glu
```

In [19]:
```
# Boxplot del IMC
plt.boxplot(df['BMI'].dropna())
plt.title('Boxplot del Índice de Masa Corporal (BMI)')
plt.ylabel('BMI')
plt.show()
```



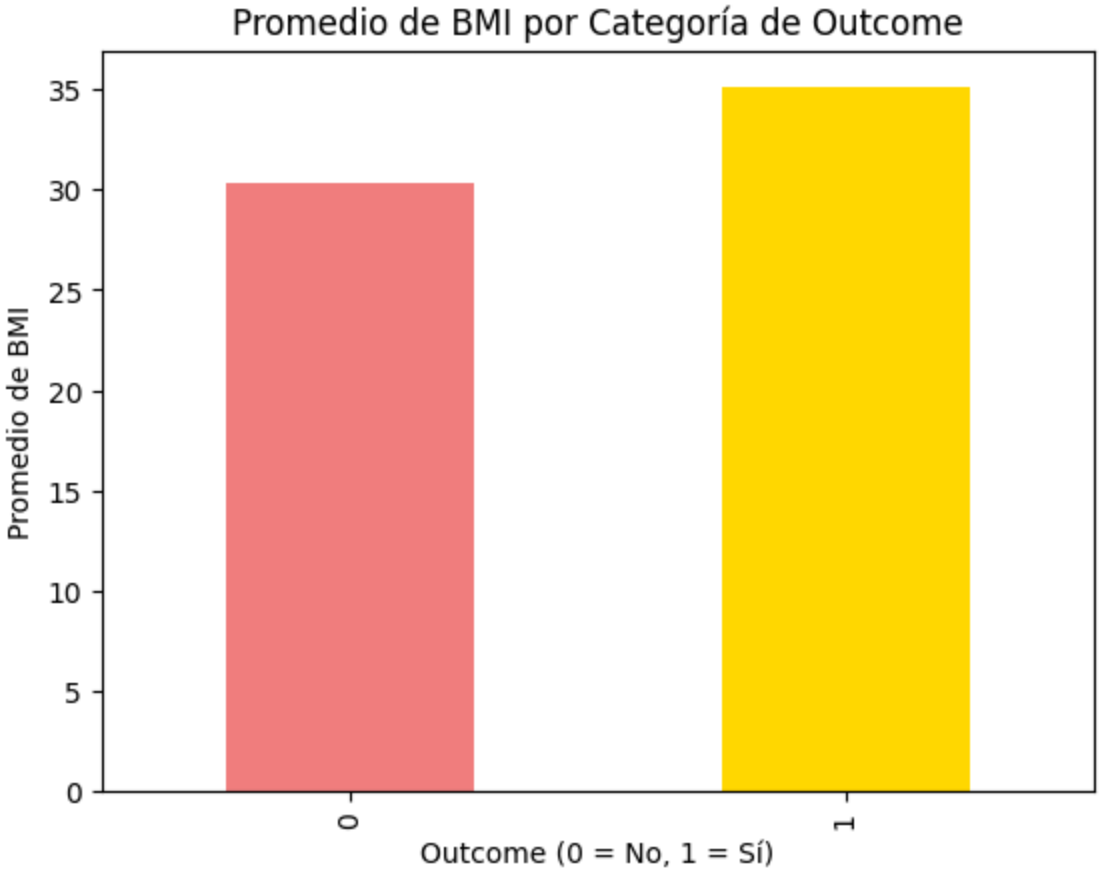
In [ ]:
```
El diagrama muestra que la mayoría de los valores del BMI estan ubicados ent
```

In [25]:
```
# Promedio de Glucose según diagnóstico de diabetes
plt.figure(figsize=(6,5))
sns.boxplot(x='Outcome', y='Glucose', data=df, hue='Outcome', dodge=False, p
sns.stripplot(x='Outcome', y='Glucose', data=df, color='orange', alpha=0.6)
plt.title('Niveles de Glucosa de acuerdo al diagnóstico de diabetes')
plt.xlabel('Outcome (0 = No, 1 = Sí)')
plt.ylabel('Glucose')
plt.show()
```



In [ ]:
```
La grafica mmuestra que las personas que fueron diagnosticados con diabetes
```

In [21]:
```
# Promedio de BMI según diagnóstico de diabetes
prom_bmi = df.groupby('Outcome')['BMI'].mean()
prom_bmi.plot(kind='bar', color=['lightcoral', 'gold'])
plt.title('Promedio de BMI por Categoría de Outcome')
plt.xlabel('Outcome (0 = No, 1 = Sí)')
plt.ylabel('Promedio de BMI')
plt.show()
```



In [ ]:
```
La gráfica muestra que las personas con diabetes tienen un promedio de BMI m
```

In [30]:
```
variables_numericas = df[['Glucose', 'BMI', 'Outcome']]
matriz_correlacion = variables_numericas.corr().round(2)
```

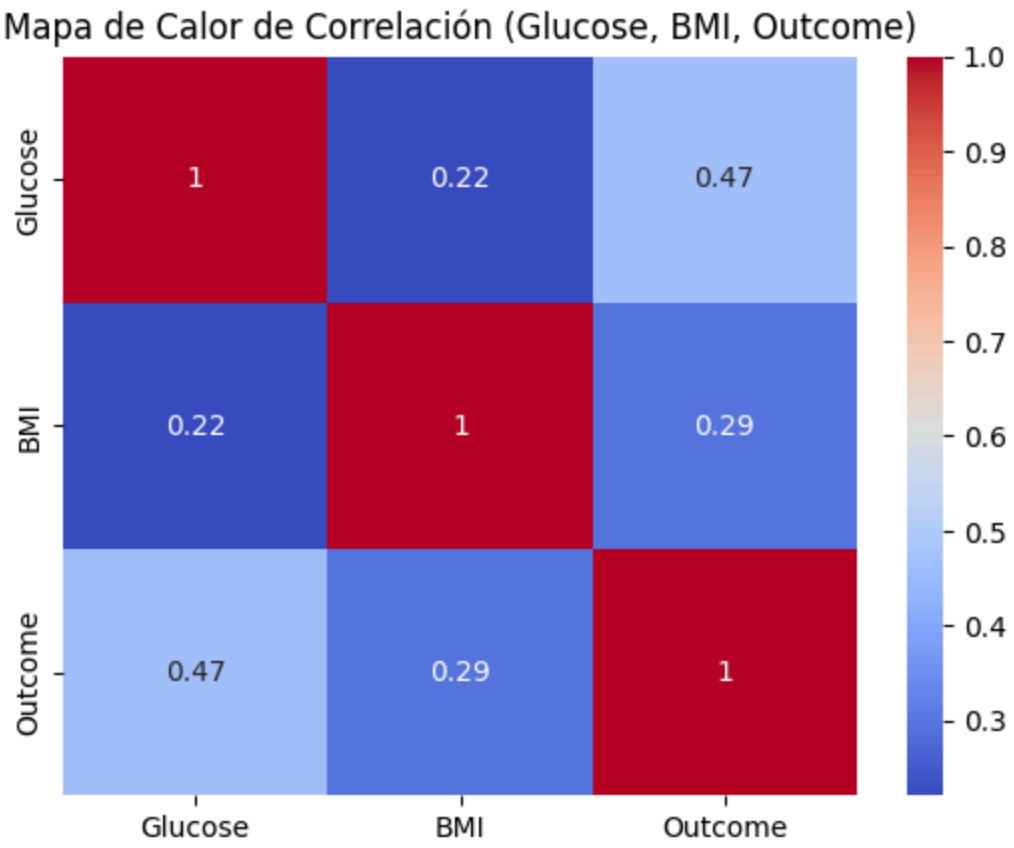
In [31]:
```
matriz_correlacion
```

Out[31]:

	Glucose	BMI	Outcome
Glucose	1.00	0.22	0.47
BMI	0.22	1.00	0.29
Outcome	0.47	0.29	1.00

In [ ]:
```
La tabla muestra que el nivel de glucosa está más relacionado con tener diab
```

In [22]:
```
# Mapa de calor de correlación entre las variables principales
correlacion = df[['Glucose', 'BMI', 'Outcome']].corr()
sns.heatmap(correlacion, annot=True, cmap='coolwarm')
plt.title('Mapa de Calor de Correlación (Glucose, BMI, Outcome)')
plt.show()
```



In [ ]:
```
El mapa de calor demuestra que hay una conexión normal entre la glucosa y el
```