Estadística Descriptiva — Pima Indians Diabetes

Autor: Fabiola Ochoa y Aaron Cuevas · **Fecha:** 2025-10-31

Objetivo de la actividad

Replicar los pasos de salón sobre el dataset `diabetes.csv` (**Pima Indians Diabetes**):

1. **Cargar** los datos.
2. Verificar **cantidad de datos** (filas/columnas) y **nombres de variables**.
3. Revisar **tipos de dato** e **identificar valores nulos**.
4. **Seleccionar** tres variables por integrante; aquí se analizan **Pregnancies**, **DiabetesPedigreeFunction** y **Outcome** (común para todos).
5. Para cada variable seleccionada: **tipo/subtipo**, **rangos (mín–máx)**, **media**, **mediana**, **desviación estándar (DE)** y **comentarios**.
6. Realizar **3 consultas** sobre los datos con las variables asignadas.

1. Carga de datos

In [1]:
```
import pandas as pd, numpy as np
from pathlib import Path

candidates = [Path("data/diabetes.csv"), Path("diabetes.csv")]
for p in candidates:
    if p.exists():
        CSV_PATH = p
        break
else:
    raise FileNotFoundError("No se encontró diabetes.csv.")

df = pd.read_csv(CSV_PATH)

# Ceros imposibles como NA en columnas clínicas estándar
cols_zero_na = [c for c in ["Glucose","BloodPressure","SkinThickness","Insul
df[cols_zero_na] = df[cols_zero_na].replace(0, np.nan)

df.head()
```

Out[1]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigr
0	6	148.0	72.0	35.0	NaN	33.6	
1	1	85.0	66.0	29.0	NaN	26.6	
2	8	183.0	64.0	NaN	NaN	23.3	
3	1	89.0	66.0	23.0	94.0	28.1	
4	0	137.0	40.0	35.0	168.0	43.1	

2. Cantidad de datos y variables

In [2]:
```
df.shape, df.columns.tolist()
```

Out[2]:
```
((768, 9),
 ['Pregnancies',
  'Glucose',
  'BloodPressure',
  'SkinThickness',
  'Insulin',
  'BMI',
  'DiabetesPedigreeFunction',
  'Age',
  'Outcome'])
```

3. Información general y valores nulos

In [3]:
```
df.info()
print("\nValores nulos por columna:")
df.isna().sum()
```

```
<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                   763 non-null    float64
 2   BloodPressure             733 non-null    float64
 3   SkinThickness             541 non-null    float64
 4   Insulin                   394 non-null    float64
 5   BMI                       757 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(6), int64(3)
memory usage: 54.1 KB

Valores nulos por columna:
```

Out[3]:
```
Pregnancies                 0
Glucose                     5
BloodPressure              35
SkinThickness             227
Insulin                   374
BMI                        11
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

4. Variables seleccionadas y tipología

- **Pregnancies**: *cuantitativa discreta* (conteo de embarazos; 0 es posible).
- **DiabetesPedigreeFunction (DPF)**: *cuantitativa continua* (índice de antecedentes familiares).
- **Outcome**: *categórica binaria* (0 = no diabetes, 1 = diabetes). Su media equivale a la prevalencia muestral.

5. Estadísticos descriptivos

In [13]:
```
sel = ["BloodPressure","Insulin","Outcome"]
stats = df.agg({
    "BloodPressure": ["min","max","mean","median","std"],
    "Insulin": ["min","max","mean","median","std"],
    "Outcome": ["min","max","mean"]
})
```

```
stats
```

Out[13]:

	BloodPressure	Insulin	Outcome
min	24.000000	14.000000	0.000000
max	122.000000	846.000000	1.000000
mean	72.405184	155.548223	0.348958
median	72.000000	125.000000	NaN
std	12.382158	118.775855	NaN

In [14]:

```
def iqr(s):
    return s.quantile(0.75) - s.quantile(0.25)
pd.DataFrame({
    "BloodPressure_IQR":[iqr(df["BloodPressure"])],
    "Insulin_IQR":[iqr(df["Insulin"])]
})
```

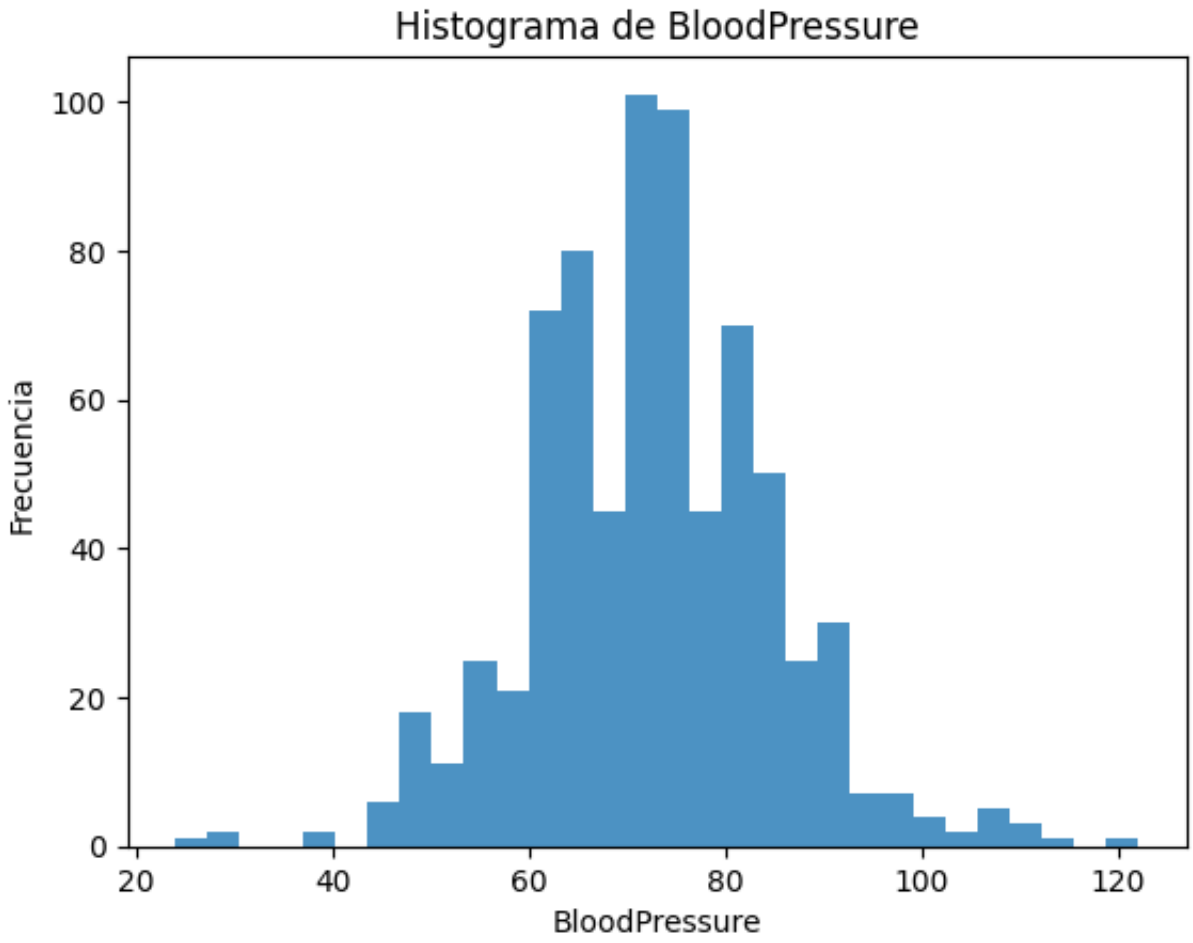
Out[14]:

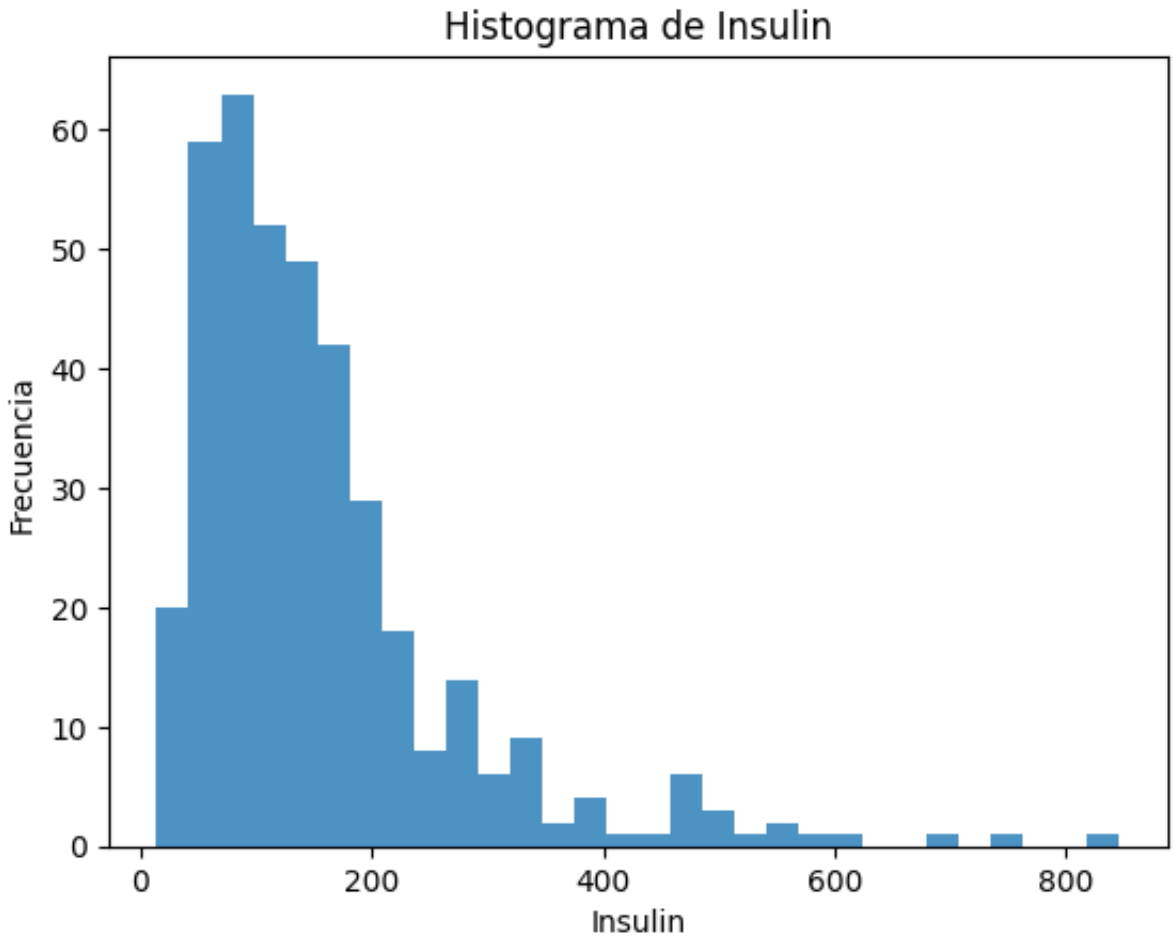
	BloodPressure_IQR	Insulin_IQR
0	16.0	113.75

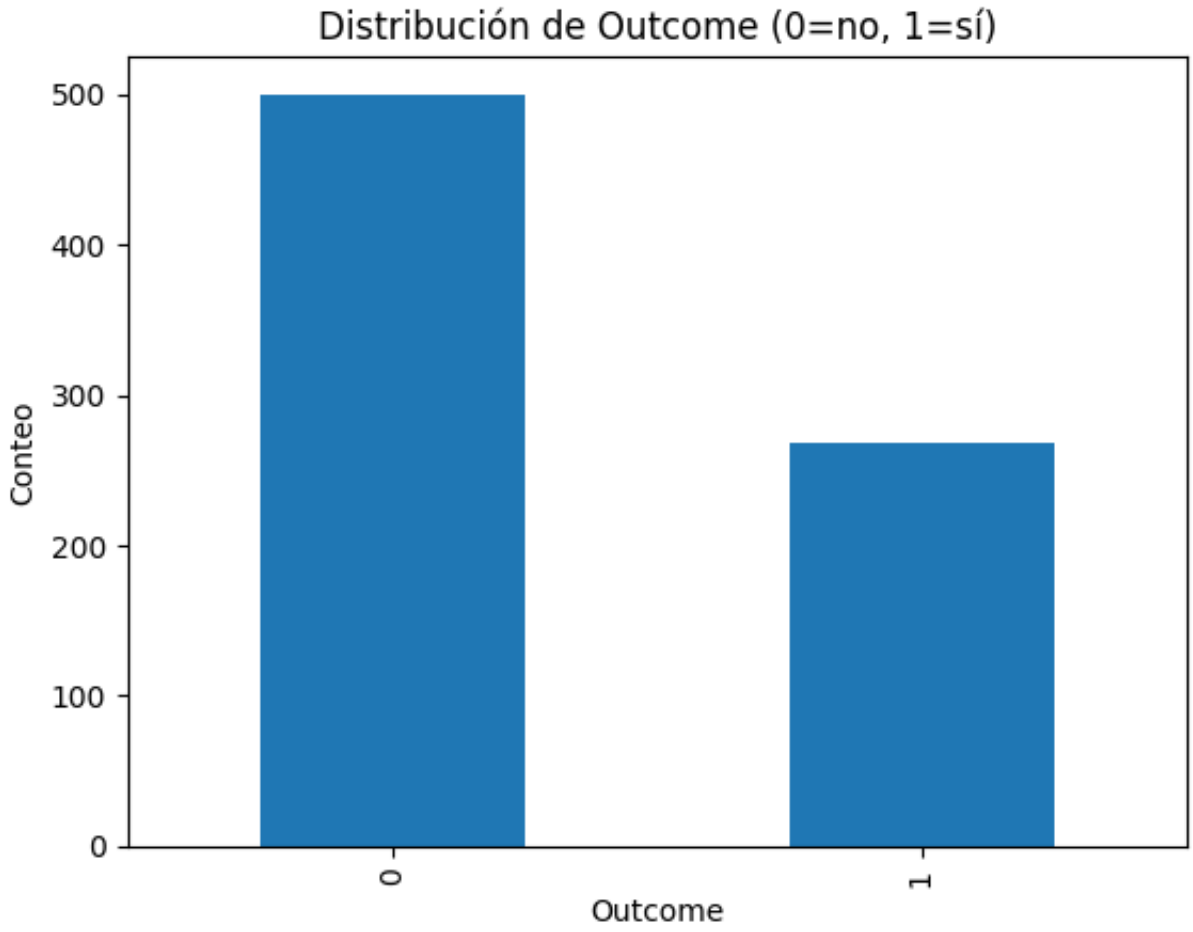
6. Visualización

In [28]:

```
import matplotlib.pyplot as plt
df["BloodPressure"].dropna().plot(kind="hist", bins=30, alpha=0.8, title="Hi
plt.xlabel("BloodPressure"); plt.ylabel("Frecuencia"); plt.show()

df["Insulin"].dropna().plot(kind="hist", bins=30, alpha=0.8, title="Histogra
plt.xlabel("Insulin"); plt.ylabel("Frecuencia"); plt.show()

df["Outcome"].value_counts().sort_index().plot(kind="bar", title="Distribuci
plt.xlabel("Outcome"); plt.ylabel("Conteo"); plt.show()
```


Histograma de BloodPressure
Frecuencia
BloodPressure
0
20
40
60
80
100
20
40
60
80
100
120


Histograma de Insulin
Frecuencia
0
10
20
30
40
50
60
0
200
400
600
800
Insulin



7. Consultas

In [16]:
```
# Q1: Pacientes con ≥5 embarazos y Outcome=1
q1 = (df.query("Pregnancies >= 5 and Outcome == 1")
        [["BloodPressure","Insulin","Outcome"]]
        .sort_values(["BloodPressure","Insulin"], ascending=False))
q1.head(10)
```

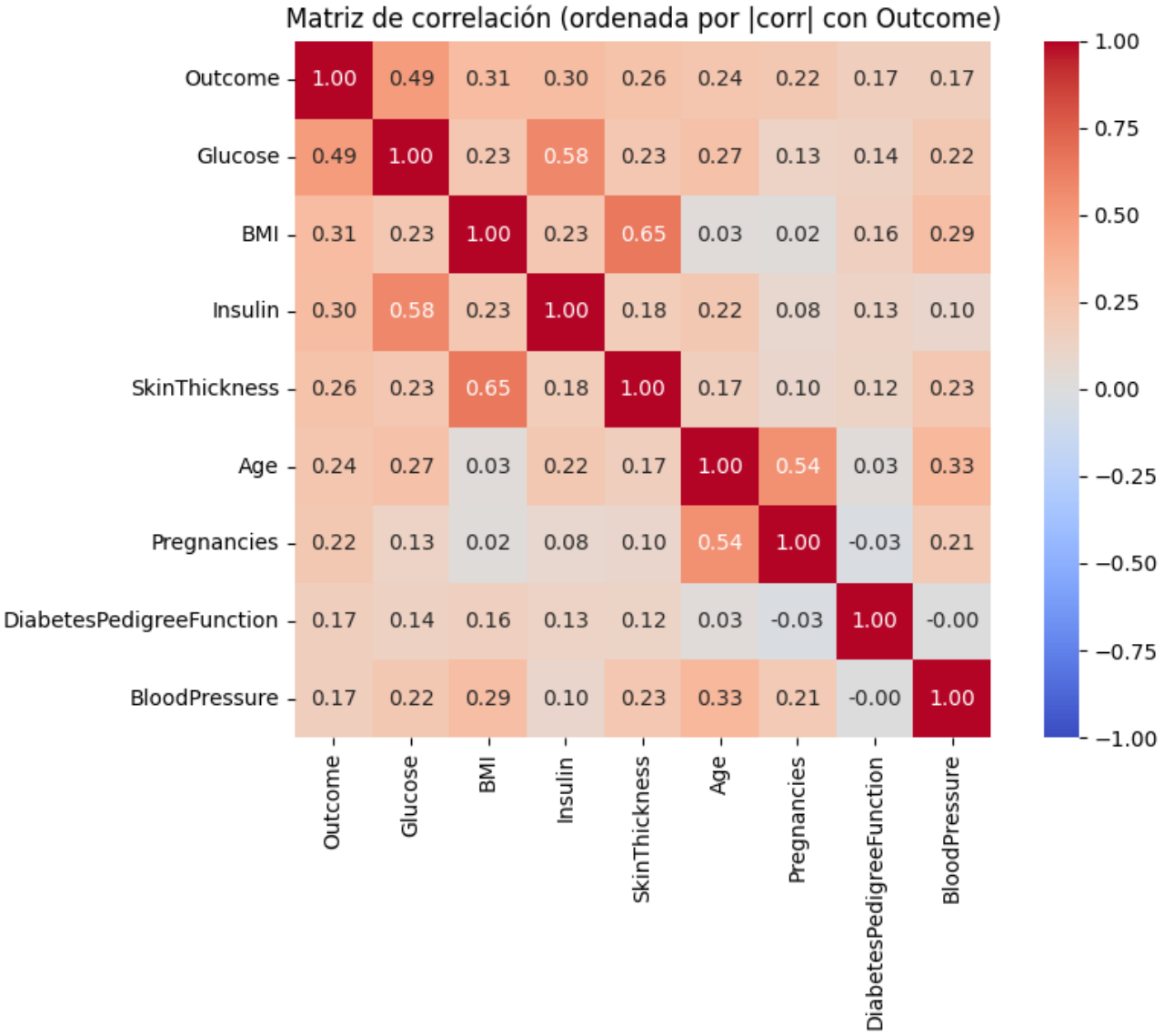
Out[16]:

	BloodPressure	Insulin	Outcome
691	114.0	NaN	1
43	110.0	240.0	1
84	108.0	NaN	1
662	106.0	231.0	1
207	104.0	NaN	1
387	100.0	NaN	1
303	98.0	NaN	1
9	96.0	NaN	1
24	94.0	146.0	1
743	94.0	NaN	1

In [23]:

```
import pandas as pd, numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Carga + limpieza mínima
try:
    df
except NameError:
    df = pd.read_csv("data/diabetes.csv")

for c in ["Glucose","BloodPressure","SkinThickness","Insulin","BMI"]:
    if c in df.columns:
        df[c] = df[c].replace(0, np.nan)

# Matriz de correlación (numéricas)
corr = df.corr(numeric_only=True)

# Orden por |corr con Outcome|, poniendo Outcome al frente
if "Outcome" not in corr.columns:
    raise ValueError("Outcome no es numérica o no existe en df.")

order = (
    ["Outcome"] +
    corr["Outcome"]
        .drop(labels=["Outcome"])
        .abs()
        .sort_values(ascending=False)
        .index
        .tolist()
```

```
)

# Reordenar filas y columnas con el MISMO orden → matriz cuadrada con diagon
corr_ord = corr.loc[order, order]

# Heatmap cuadrado con diagonal en 1 (Outcome en la esquina superior izquier
plt.figure(figsize=(9, 7))
sns.heatmap(
    corr_ord.round(2),
    annot=True, fmt=".2f",
    vmin=-1, vmax=1, cmap="coolwarm",
    square=True, cbar=True
)
plt.title("Matriz de correlación (ordenada por |corr| con Outcome)")
plt.tight_layout()
plt.show()
```



In [12]:
```
# Q3: Pr(Outcome=1) por cuartiles de DPF
q3 = (df.assign(dpf_q=pd.qcut(df["DiabetesPedigreeFunction"], q=4, duplicate
        .groupby("dpf_q")["Outcome"].mean()
        .rename("Pr(Outcome=1)")).to_frame()
q3
```

```
/var/folders/8g/s9mnwlbj22v615wt4ylwhq_c0000gn/T/ipykernel_20131/1631812449.
py:2: FutureWarning: The default of observed=False is deprecated and will be
changed to True in a future version of pandas. Pass observed=False to retain
current behavior or observed=True to adopt the future default and silence th
is warning.
  q3 = (df.assign(dpf_q=pd.qcut(df["DiabetesPedigreeFunction"], q=4, duplica
tes="drop"))
```

Out[12]:

dpf_q	Pr(Outcome=1)
(0.077, 0.244]	0.255208
(0.244, 0.372]	0.333333
(0.372, 0.626]	0.322917
(0.626, 2.42]	0.484375

8. Conclusiones

a) Tipos y limpieza. `Pregnancies` es **discreta** (conteo), `DiabetesPedigreeFunction` es **continua** (índice) y `Outcome` es **binaria** (0/1). Los ceros imposibles se trataron como faltantes en variables clínicas no analizadas aquí (Glucose, BloodPressure, SkinThickness, Insulin, BMI), evitando sesgos en estadísticas agregadas.

b) Rango y tendencia central.

- `Pregnancies` : **mín 0**, **máx 17**, **media 3.85**, **mediana 3.00**, **DE 3.37**, **IQR 5.00**. La **media ≥ mediana** sugiere **asimetría a la derecha**: abundan valores bajos con una cola de conteos altos.
- `DiabetesPedigreeFunction` : **mín 0.078**, **máx 2.420**, **media 0.472**, **mediana 0.372**, **DE 0.331**, **IQR 0.382**. También asimétrica a la derecha: muchos valores pequeños y algunos grandes.
- `Outcome` : **prevalencia muestral ≈ 34.9%** (la media de `Outcome`).

c) Asociación empírica con Outcome.

- Por **categorías de embarazos** (`Pr(Outcome=1)`):
- 0: 0.272
- 1–2: 0.258
- 3–4: 0.352
- 5–9: 0.492
- 10+: 0.588 Se observa un **patrón creciente**: a mayor número de embarazos, mayor proporción de casos positivos.
- Por **cuartiles de DPF** (`Pr(Outcome=1)`):
- (0.077, 0.244]: 0.255
- (0.244, 0.372]: 0.333
- (0.372, 0.626]: 0.323
- (0.626, 2.42]: 0.484 La proporción **aumenta de cuartil en cuartil**, coherente con que valores altos del índice familiar se asocian a mayor riesgo.

d) Lectura integrada y límites.

- Ambas variables (`Pregnancies` , `DPF`) muestran relación **monótona creciente** con `Outcome` , pero de **magnitud moderada** (fenómeno multifactorial).
- Posibles **confusores** (no modelados aquí): edad, IMC y glucosa; podrían explicar parte de los incrementos observados.
- La prevalencia reportada es **muestral**; no debe interpretarse como tasa poblacional sin muestreo representativo.

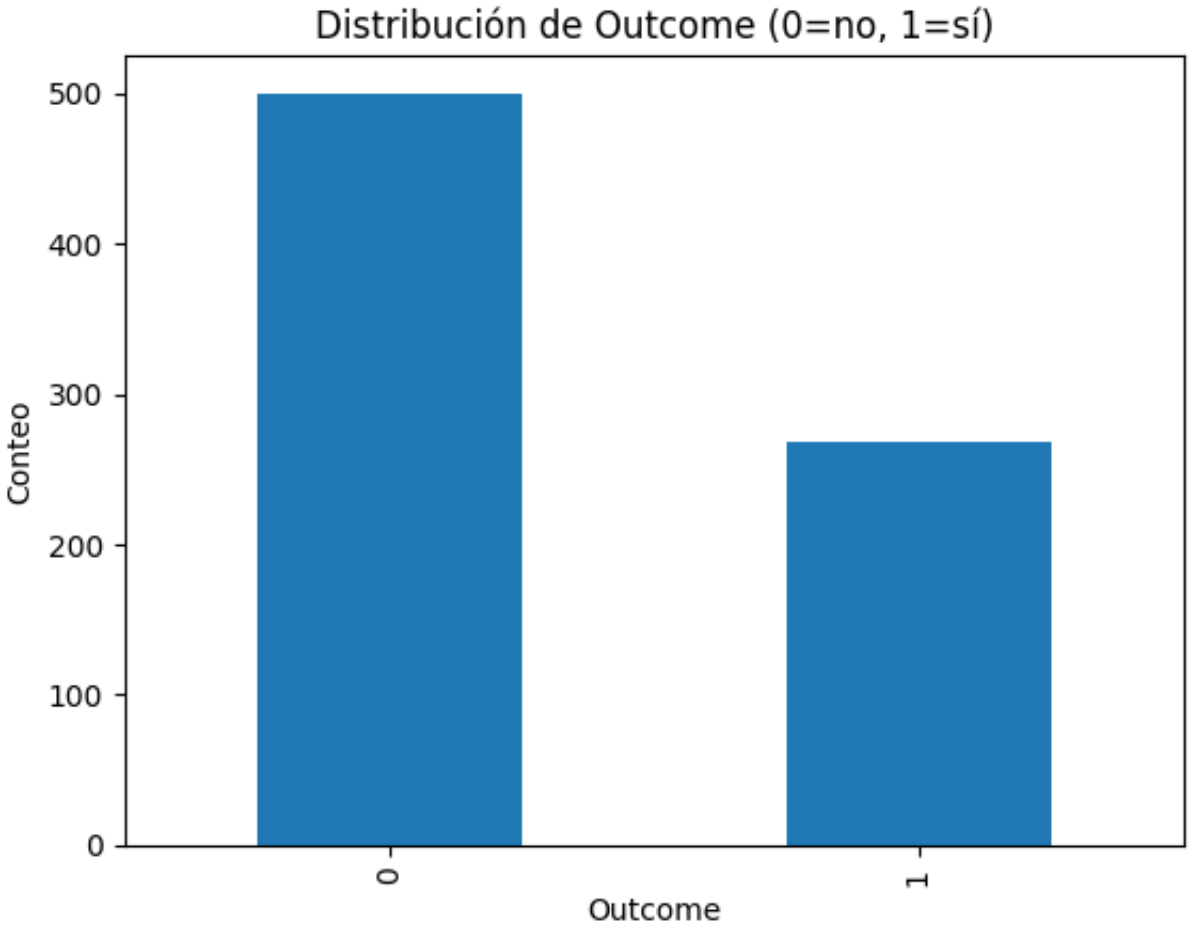
e) Recomendación inmediata.

- Mantener estas variables en un modelo logístico base `Outcome ~ Pregnancies + DPF` y, si procede, ajustar por edad, IMC y glucosa para estimar efectos **parciales**.

Visualización y Análisis de Datos

Variables categóricas

In [21]:
```
import matplotlib.pyplot as plt

# Barras para Outcome (única categórica estricta)
datos_outcome = df['Outcome'].value_counts().sort_index()
datos_outcome.plot(kind='bar')
plt.title('Distribución de Outcome (0=no, 1=sí)')
plt.xlabel('Outcome'); plt.ylabel('Conteo')
plt.show()
```



In [27]:
```
import pandas as pd, numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
try:
    df
except NameError:
    import pathlib
    df = pd.read_csv(pathlib.Path('data/diabetes.csv'))
    for c in ['Glucose','BloodPressure','SkinThickness','Insulin','BMI']:
        if c in df.columns: df[c]=df[c].replace(0, np.nan)
```