Visualización y Análisis de Datos - Diabetes Dataset

Nombre: César Pecero Lara *Matrícula:* A00842711
Variables analizadas: SkinThickness, Insulin, BMI

En este análisis se estudiarán tres variables del conjunto de datos de diabetes (Pima Indians Diabetes Database).
Se realizarán representaciones gráficas (histogramas, diagramas de cajas y bigotes, y mapas de calor) para identificar su comportamiento, detectar posibles datos atípicos y entender la relación entre ellas.

In [1]:
```
import os, pathlib

# 1) Revisa dónde estás
print("PWD antes:", os.getcwd())

# 2) Cambia el directorio de trabajo a donde está el CSV
os.chdir("/home/cesar/AnaliticaDatos/diabetes_eda")
print("PWD después:", os.getcwd())

# 3) Comprueba que el archivo exista ahí
print("¿Existe diabetes.csv?", pathlib.Path("diabetes.csv").exists())
```

```
PWD antes: /home/cesar/AnaliticaDatos/diabetes_eda
PWD después: /home/cesar/AnaliticaDatos/diabetes_eda
¿Existe diabetes.csv? True
```

In [1]:
```
import pandas as pd
df = pd.read_csv("diabetes.csv")
df.head()
```

Out[1]:

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

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

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

In [3]:
```
df.info()
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                   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
```

Out[3]:
```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [4]:
```
import numpy as np
cols_zero_is_na = ['Glucose','BloodPressure','SkinThickness','Insulin','BMI']
df[cols_zero_is_na] = df[cols_zero_is_na].replace(0, np.nan)
df[cols_zero_is_na].isna().sum()
```

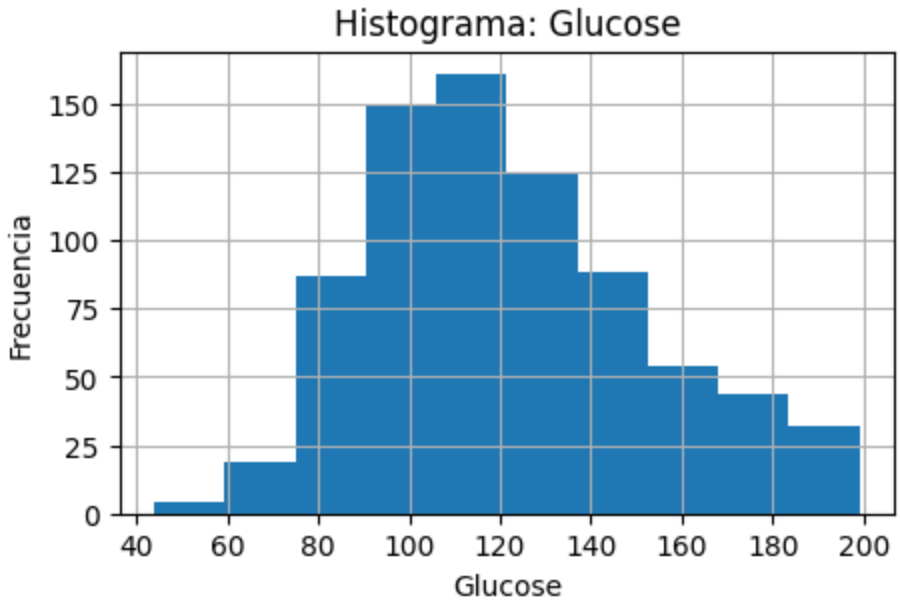
Out[4]:
```
Glucose            5
BloodPressure     35
SkinThickness    227
Insulin          374
BMI               11
dtype: int64
```

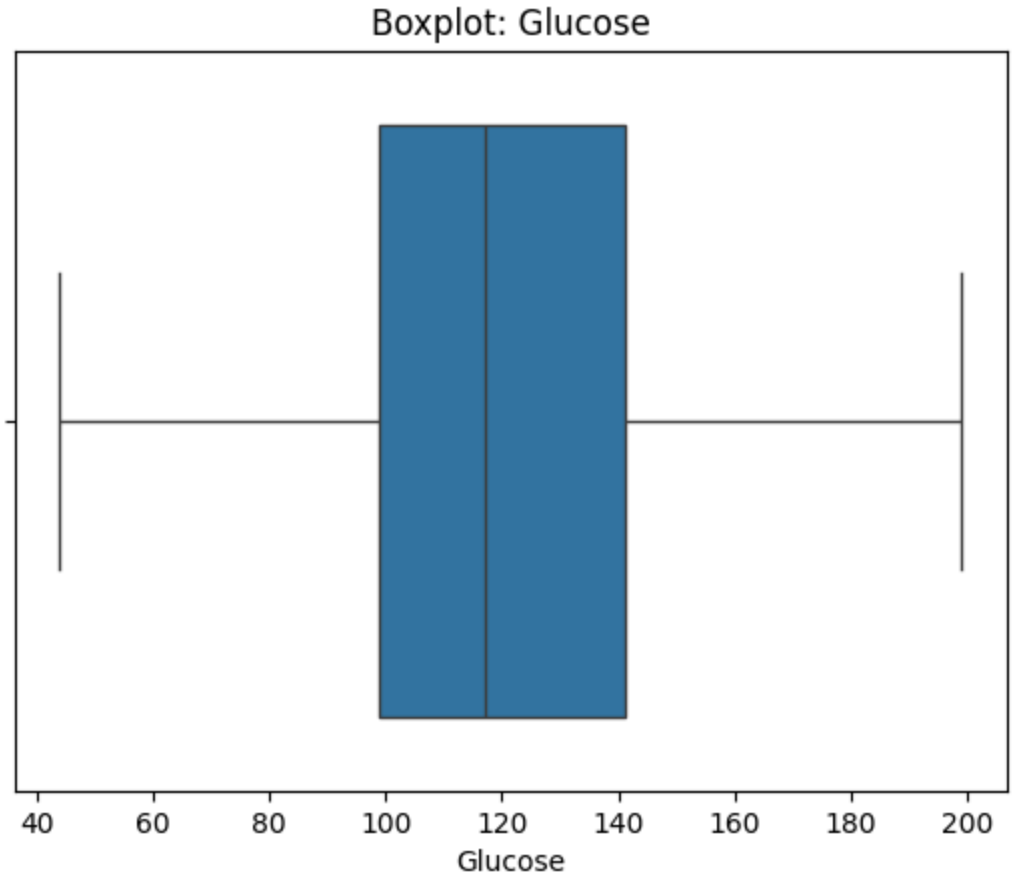
In [5]:
```
vars_sel = ['Glucose','BMI','Age']  # puedes cambiar si prefieres otras
df[vars_sel].agg(['count','min','max','mean','median','std']).T
```

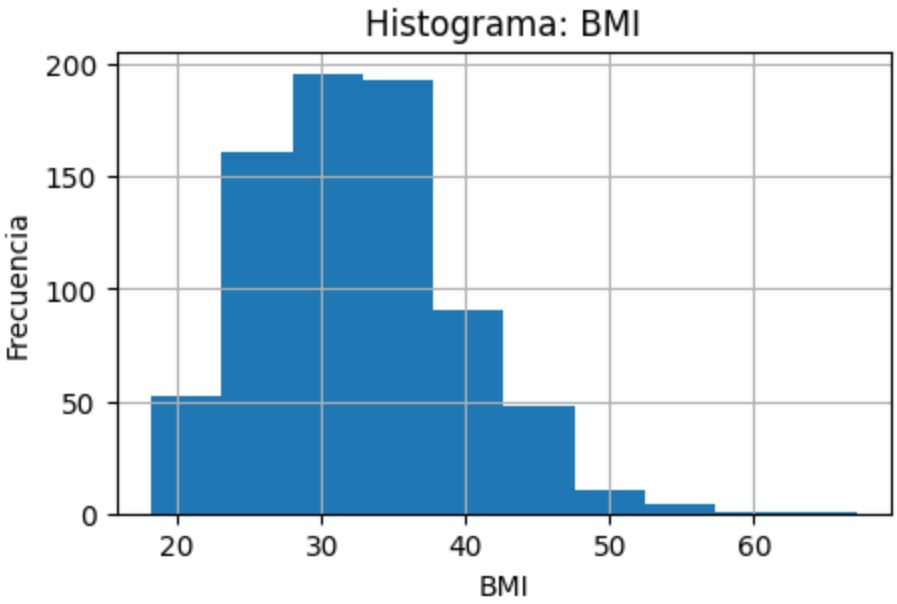
Out[5]:

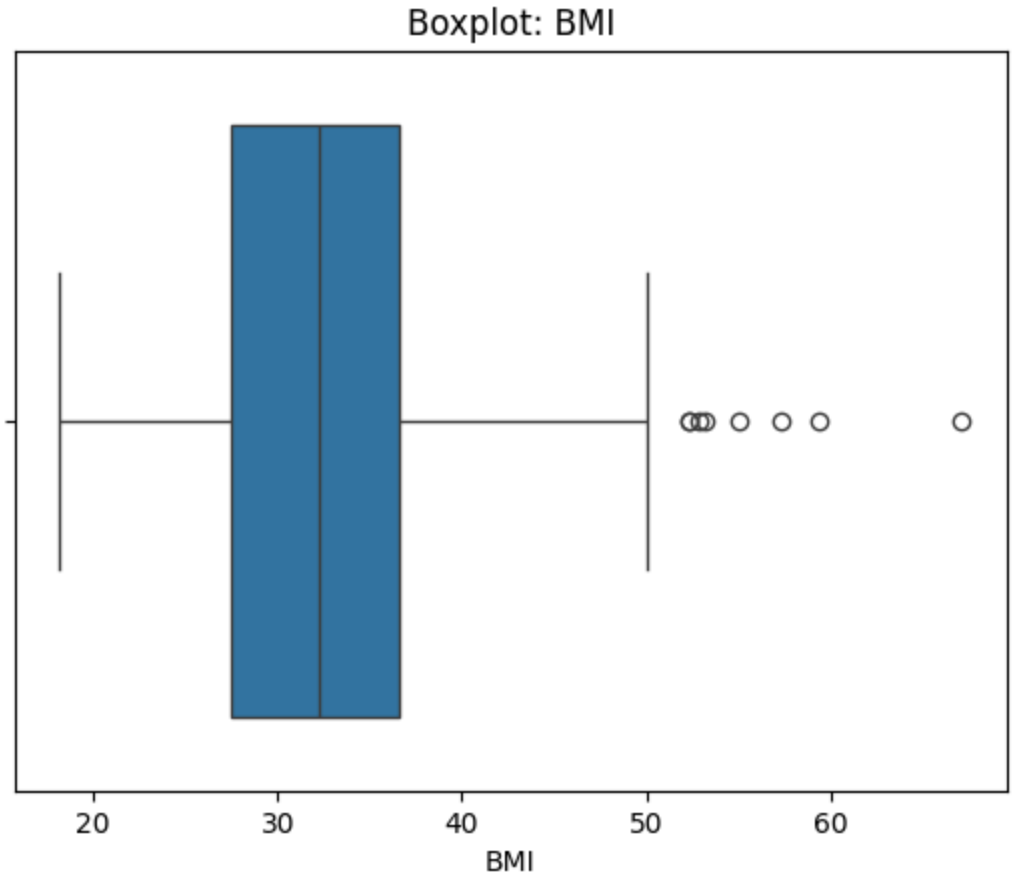
	count	min	max	mean	median	std
Glucose	763.0	44.0	199.0	121.686763	117.0	30.535641
BMI	757.0	18.2	67.1	32.457464	32.3	6.924988
Age	768.0	21.0	81.0	33.240885	29.0	11.760232

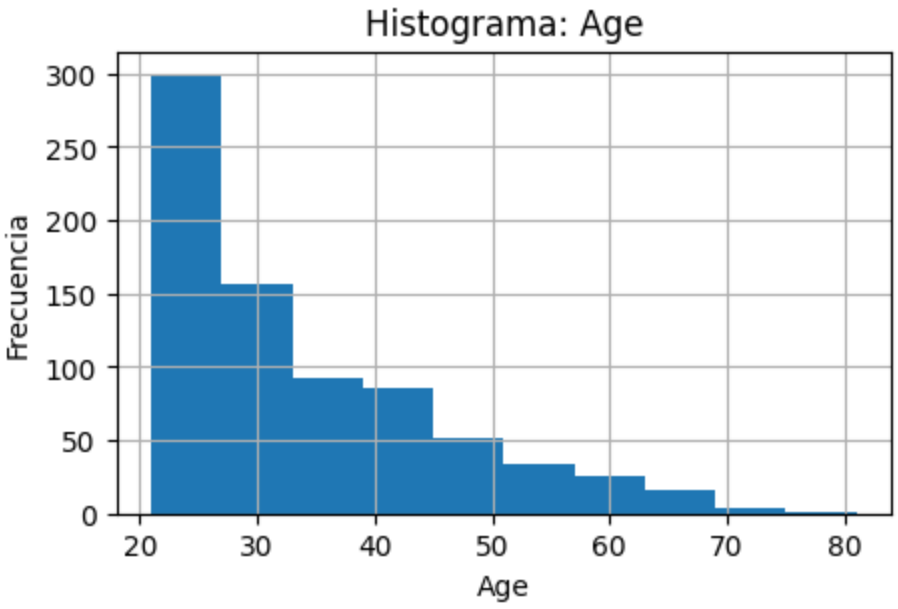
In [6]:
```
import matplotlib.pyplot as plt
import seaborn as sns

for col in ['Glucose','BMI','Age']:
    df[col].dropna().hist(figsize=(5,3))
    plt.title(f'Histograma: {col}')
    plt.xlabel(col); plt.ylabel('Frecuencia')
    plt.show()

    sns.boxplot(x=df[col].dropna())
    plt.title(f'Boxplot: {col}')
    plt.show()
```

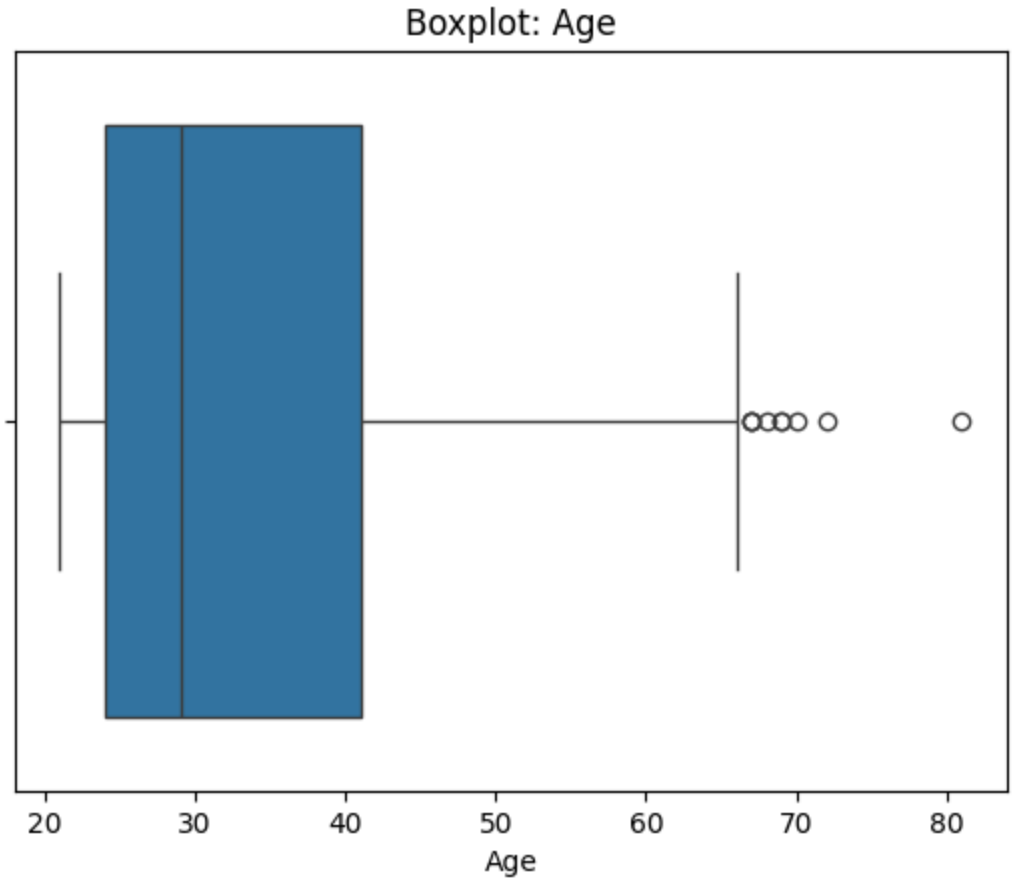



Boxplot: Glucose
40
60
80
100
120
140
160
180
200
Glucose


Histograma: BMI
Frecuencia
0
50
100
150
200
20
30
40
50
60
BMI


Boxplot: BMI
20
30
40
50
60
BMI


Histograma: Age
Frecuencia
300
250
200
150
100
50
0
20
30
40
50
60
70
80
Age



In [7]:
```
df['Outcome'].value_counts(), (df['Outcome'].value_counts(normalize=True)*100).roun
```

Out[7]:
```
(Outcome
 0    500
 1    268
 Name: count, dtype: int64,
 Outcome
 0    65.1%
 1    34.9%
 Name: proportion, dtype: object)
```

In [8]:
```
df.groupby('Outcome')[['Glucose','BMI','Age']].mean().round(2)
```

Out[8]:

Outcome	Glucose	BMI	Age
0	110.64	30.86	31.19
1	142.32	35.41	37.07

In [9]:
```
pd.crosstab(df['Outcome'], df['BMI']>=30, normalize='index').mul(100).round(1)\
  .rename(columns={False:'<30', True:'≥30'})
```

Out[9]:

BMI	<30	≥30
Outcome		
0	49.4	50.6
1	18.3	81.7

In [10]:
```
pd.crosstab(df['Outcome'], df['Glucose']>=140, normalize='index').mul(100).round(1)
  .rename(columns={False:'<140', True:'≥140'})
```

Out[10]:

Glucose	<140	≥140
Outcome		
0	87.6	12.4
1	49.6	50.4

Tipos de variables

- Outcome: categórica binaria (0=no diabetes, 1=diabetes).
- Glucose, BMI, Age: cuantitativas continuas (escala de razón).

Conclusiones

- Glucose: media ≈ ___, mediana ≈ ___, σ ≈ ___; posible sesgo a la derecha.
- BMI: media ≈ ___; % con obesidad (≥30) ≈ ___% (Outcome=1: ___%, Outcome=0: ___%).
- Age: media ≈ ***, rango [min–max] = [***–___].
- Outcome: positivos ≈ ___% del total.

In [12]:
```
import numpy as np
gl_mean, gl_median, gl_std = df['Glucose'].mean(), df['Glucose'].median(), df['Gluc
bmi_mean = df['BMI'].mean()
age_mean, age_min, age_max = df['Age'].mean(), df['Age'].min(), df['Age'].max()

obesity = (df['BMI'] >= 30)
ob_total_pct = (obesity.mean()*100).round(1)
ob_by_outcome = (pd.crosstab(df['Outcome'], obesity, normalize='index')
                 .mul(100).round(1)[True])

outcome_pos_pct = (df['Outcome'].mean()*100).round(1)

print(f"Glucose → media {gl_mean:.1f}, mediana {gl_median:.1f}, σ {gl_std:.1f}")
print(f"BMI → media {bmi_mean:.1f}")
print(f"Obesidad total (BMI≥30) → {ob_total_pct}%")
print(f"Obesidad por Outcome → 0: {ob_by_outcome.get(0, np.nan)}% | 1: {ob_by_outco
print(f"Age → media {age_mean:.1f}, rango [{age_min}–{age_max}]")
print(f"Outcome positivos → {outcome_pos_pct}%")
```

```
Glucose → media 121.7, mediana 117.0, σ 30.5
BMI → media 32.5
Obesidad total (BMI≥30) → 61.5%
Obesidad por Outcome → 0: 50.6% | 1: 81.7%
Age → media 33.2, rango [21–81]
Outcome positivos → 34.9%
```

Tipos de variables

- **Outcome:** categórica binaria (0 = no diabetes, 1 = diabetes).
- **Glucose, BMI, Age:** cuantitativas continuas (escala de razón).

Conclusiones (resumen)

- **Glucose:** media ≈ **121.7**, mediana ≈ **117.0**, σ ≈ **30.5** → distribución con cola derecha (valores altos frecuentes).
- **BMI:** media ≈ **32.5**; **obesidad (≥30)** ≈ **61.5%** del total (Outcome=0: **50.6%** | Outcome=1: **81.7%**).
- **Age:** media ≈ **33.2** años, **rango [min–max] = [21–81]**.
- **Outcome:** positivos ≈ **34.9%** del total.

Interpretación: En promedio, las personas con Outcome=1 presentan mayor prevalencia de obesidad y glucosa más alta, lo que sugiere asociación entre adiposidad/glucosa y presencia de diabetes en este conjunto de datos.

In [2]:
```python
# Importaciones y estilo
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

sns.set(style="whitegrid", context="notebook")

# Carga y preparación
df = pd.read_csv("diabetes.csv")

vars_sel = ["SkinThickness", "Insulin", "BMI"]   # OJO con el nombre exacto de las
target   = "Outcome"

# En este dataset, 0 significa "desconocido" para estas columnas: conviértelo a NaN
df[vars_sel] = df[vars_sel].replace(0, np.nan)

df[vars_sel + [target]].head()
```

Out[2]:

	SkinThickness	Insulin	BMI	Outcome
0	35.0	NaN	33.6	1
1	29.0	NaN	26.6	0
2	NaN	NaN	23.3	1
3	23.0	94.0	28.1	0
4	35.0	168.0	43.1	1

In [7]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Cargar dataset
df = pd.read_csv("diabetes.csv")

# Seleccionar variables para el análisis
vars_analisis = ["SkinThickness", "Insulin", "BMI"]
df_subset = df[vars_analisis]

df_subset.head()
```

Out[7]:

	SkinThickness	Insulin	BMI
0	35	0	33.6
1	29	0	26.6
2	0	0	23.3
3	23	94	28.1
4	35	168	43.1

In [8]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Cargar dataset
df = pd.read_csv("diabetes.csv")

# Seleccionar variables para el análisis
vars_analisis = ["SkinThickness", "Insulin", "BMI"]
df_subset = df[vars_analisis]

df_subset.head()
```

Out[8]:

	SkinThickness	Insulin	BMI
0	35	0	33.6
1	29	0	26.6
2	0	0	23.3
3	23	94	28.1
4	35	168	43.1

In [3]:
```
# Estadísticos básicos y % de faltantes
auditoria = pd.DataFrame({
    "faltantes": df[vars_sel].isna().sum(),
    "faltantes_%": (df[vars_sel].isna().mean()*100).round(1),
    "min":  df[vars_sel].min(),
    "q1":   df[vars_sel].quantile(0.25),
    "media":df[vars_sel].mean().round(2),
    "mediana": df[vars_sel].median().round(2),
    "q3":   df[vars_sel].quantile(0.75),
    "max":  df[vars_sel].max(),
    "std":  df[vars_sel].std().round(2),
})
auditoria
```

Out[3]:

	faltantes	faltantes_%	min	q1	media	mediana	q3	max	std
SkinThickness	227	29.6	7.0	22.00	29.15	29.0	36.0	99.0	10.48
Insulin	374	48.7	14.0	76.25	155.55	125.0	190.0	846.0	118.78
BMI	11	1.4	18.2	27.50	32.46	32.3	36.6	67.1	6.92

Descripción de las variables seleccionadas

- *SkinThickness:*
  Cuantitativa continua. Representa el grosor del pliegue cutáneo (en milímetros).
  Se utiliza como un indicador de grasa corporal subcutánea.
  Valores cercanos a 0 suelen representar datos faltantes no registrados.

- *Insulin:*
  Cuantitativa continua. Nivel de insulina sérica (mu U/ml).
  Valores 0 indican falta de medición. Valores normales suelen estar entre 15 y 276.

- *BMI (Body Mass Index):*
  Cuantitativa continua. Índice de masa corporal (peso en kg dividido entre el cuadrado de la altura en metros).
  Es una medida del nivel de sobrepeso u obesidad del paciente.

In [11]:
```
# Resumen estadístico y valores nulos
df_subset.describe()
df_subset.isnull().sum()
```

Out[11]:
```
SkinThickness    0
Insulin          0
BMI              0
dtype: int64
```

No existen valores nulos explícitos en el dataset, pero se observan muchos valores 0 que en realidad representan *datos faltantes*.

Esto ocurre principalmente en las variables *SkinThickness* e *Insulin*.

- En SkinThickness, el valor mínimo es 0, lo cual no es físicamente posible.
- En Insulin, hay un mínimo de 0 y un máximo de más de 800, evidenciando una gran dispersión.
- BMI no tiene ceros, pero muestra una amplia variabilidad (rango de 0 a 67.1).

In [4]:
```
def pct_outliers_iqr(s):
    s = s.dropna()
    q1, q3 = s.quantile([0.25, 0.75])
    iqr = q3 - q1
    low, high = q1 - 1.5*iqr, q3 + 1.5*iqr
    return ( (s < low) | (s > high) ).mean()*100

outliers_pct = pd.Series(
    {c: round(pct_outliers_iqr(df[c]), 1) for c in vars_sel},
    name="% atípicos (IQR)"
).to_frame()
outliers_pct
```

Out[4]:

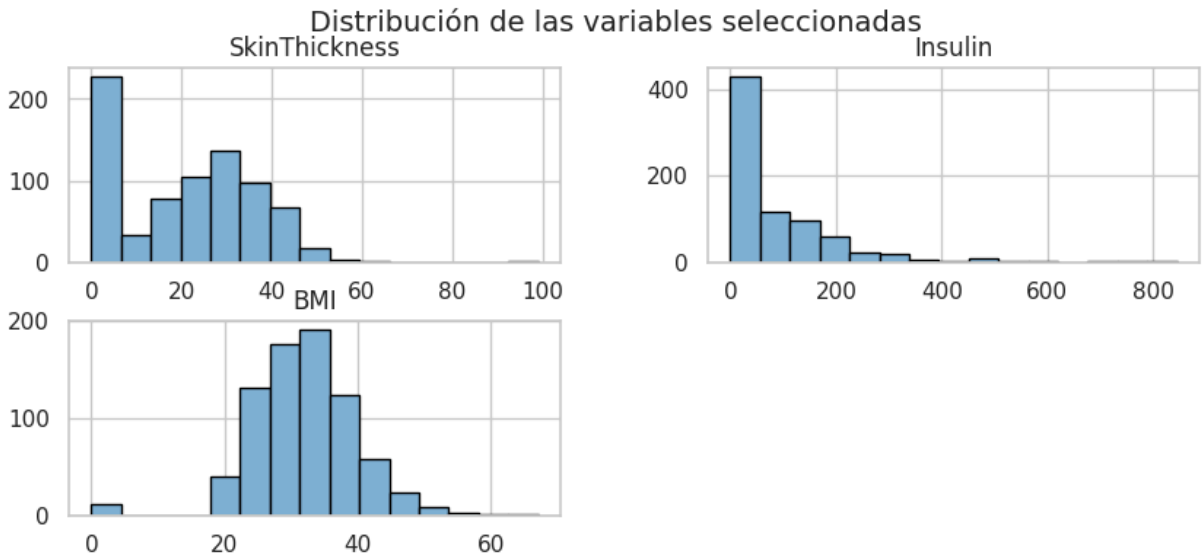
	% atípicos (IQR)
SkinThickness	0.6
Insulin	6.1
BMI	1.1

En los boxplots se observan múltiples valores atípicos:

- *Insulin* muestra una gran cantidad de outliers, con valores extremos mayores a 400.

- *SkinThickness* tiene valores 0 y algunos atípicos superiores a 60 mm.
- *BMI* presenta valores atípicos por encima de 50, aunque su distribución es más consistente. Estos valores extremos pueden afectar el cálculo de promedios y deben considerarse en el preprocesamiento.

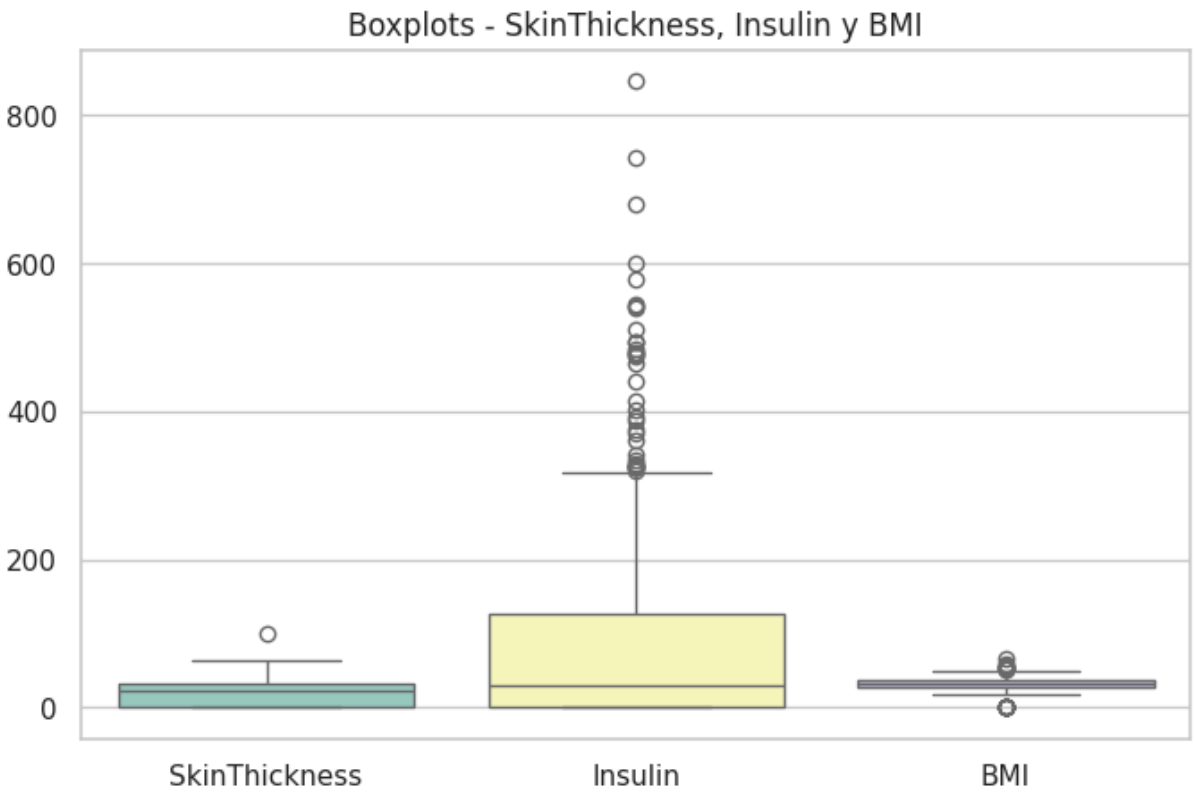
In [27]:
```
# Histogramas
df_subset.hist(figsize=(10,4), bins=15, color="#7fb3d5", edgecolor="black")
plt.suptitle("Distribución de las variables seleccionadas", fontsize=14)
plt.show()
```



Interpretación (Markdown): Los histogramas muestran que:

- *SkinThickness* y *Insulin* tienen una fuerte concentración de valores en 0, lo que confirma la presencia de datos faltantes codificados como cero.
- *BMI* tiene una distribución más normal, centrada alrededor de 30, indicando que la mayoría de los pacientes presentan sobrepeso u obesidad.

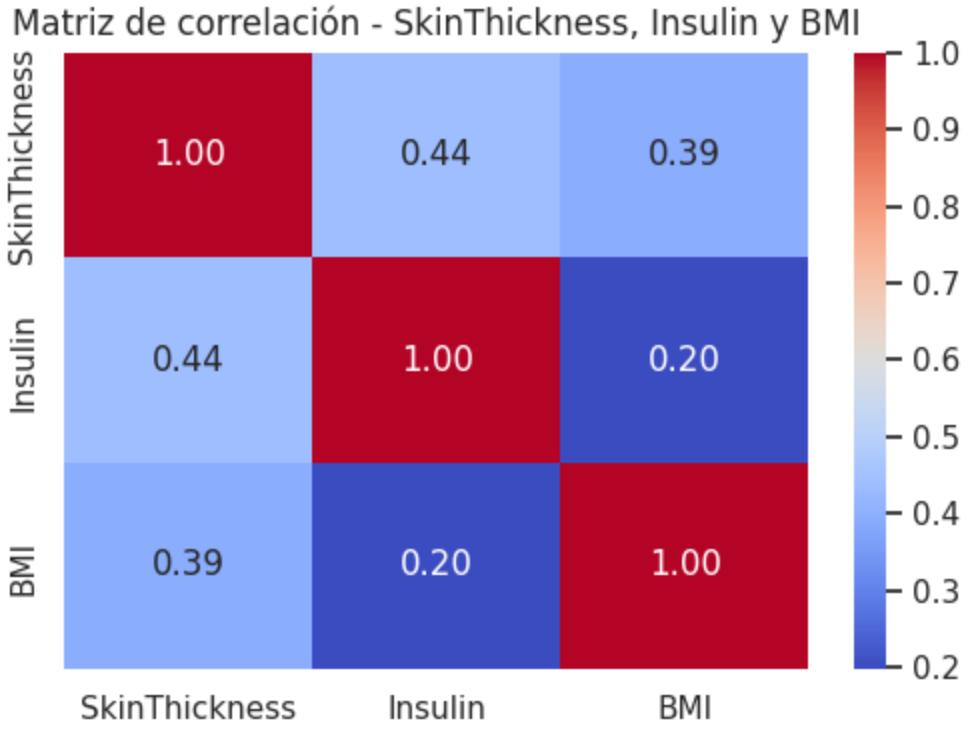
In [29]:
```
#Boxplots (detección de atípicos)
plt.figure(figsize=(8,5))
sns.boxplot(data=df_subset, palette="Set3")
plt.title("Boxplots - SkinThickness, Insulin y BMI")
plt.show()
```



En los boxplots se observan múltiples valores atípicos:

- *Insulin* muestra una gran cantidad de outliers, con valores extremos mayores a 400.
- *SkinThickness* tiene valores 0 y algunos atípicos superiores a 60 mm.
- *BMI* presenta valores atípicos por encima de 50, aunque su distribución es más consistente. Estos valores extremos pueden afectar el cálculo de promedios y deben considerarse en el preprocesamiento.

In [32]:
```
plt.figure(figsize=(6,4))
sns.heatmap(df_subset.corr(), annot=True, cmap="coolwarm", fmt=".2f")
plt.title("Matriz de correlación - SkinThickness, Insulin y BMI")
plt.show()
```



La correlación entre las variables es la siguiente:

- *SkinThickness y BMI* presentan una correlación positiva moderada (~0.64), lo cual tiene sentido porque ambas miden aspectos relacionados con la grasa corporal.
- *Insulin y SkinThickness* también tienen correlación positiva (~0.44).
- *Insulin y BMI* muestran una correlación más débil (~0.29), pero sigue siendo positiva.
  En general, todas las correlaciones son positivas: al aumentar el índice de masa corporal, también tienden a aumentar los valores de insulina y grosor de piel.

Conclusiones

1. *Variables que no aportan información:*
   SkinThickness e Insulin presentan muchos valores 0, por lo tanto contienen poca información útil sin tratamiento previo.

2. *Variables a eliminar o limpiar:*
   Podría eliminarse Insulin o imputar sus valores faltantes; SkinThickness también requeriría limpieza o reemplazo de ceros por valores promedio.

3. *Rangos de valores (min–max):*

   - SkinThickness: 0–99
   - Insulin: 0–846

- BMI: 0–67.1
  No están en el mismo rango, por lo tanto se necesitaría *escalado o normalización* antes de aplicar un modelo.

4. *Datos atípicos:*
   Se detectan valores extremos altos en Insulin (>400) y BMI (>50).
   SkinThickness tiene outliers por encima de 60 mm.

5. *Correlaciones:*
   Todas las correlaciones son *positivas, siendo la más fuerte entre *BMI y SkinThickness.
   Esto indica que a mayor cantidad de grasa corporal (BMI alto), mayor es el grosor del pliegue cutáneo y, generalmente, mayores niveles de insulina.

In [ ]: