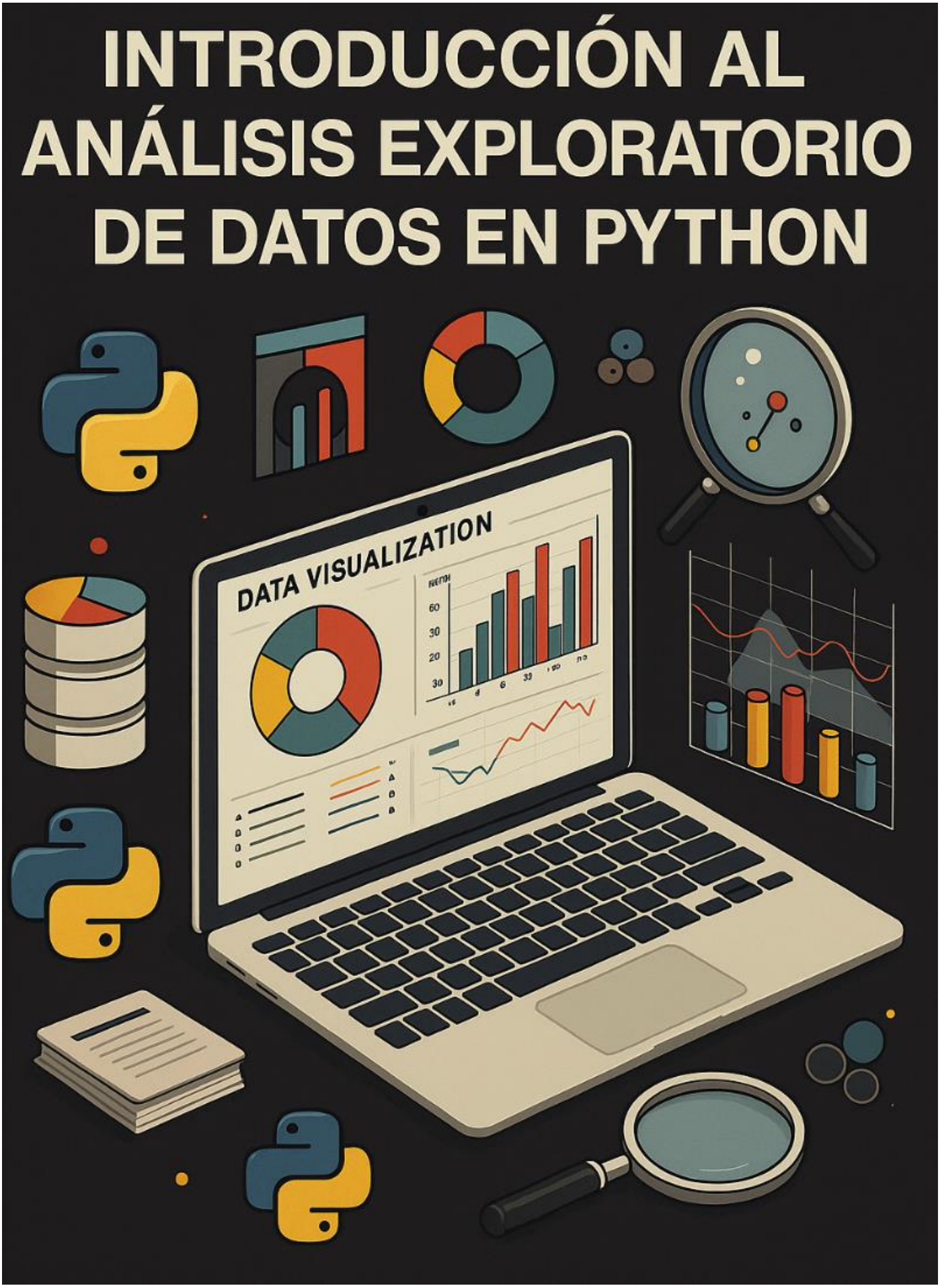
INTRODUCCIÓN AL ANÁLISIS EXPLORATORIO DE DATOS EN PYTHON



GUÍA INTRODUCTORIA AL ANÁLISIS EXPLORATORIO DE DATOS (AED) EN PYTHON

- **Autor (2025):** Juan Miguel Rodríguez Trujillo, estudiante de economía de noveno semestre
- **EconomIA:** Laboratorio de inteligencia artificial aplicada a Economía
- **Universidad Externado de Colombia, Facultad de Economia**

Introducción

En el contexto actual de la economía digital, la capacidad de analizar, comprender y comunicar patrones ocultos en los datos se ha vuelto una habilidad fundamental para economistas, investigadores y tomadores de decisiones. El Análisis Exploratorio de Datos (AED) constituye una etapa crítica en todo proceso de análisis cuantitativo, ya que permite identificar relaciones, detectar errores, visualizar distribuciones y generar hipótesis con base en los datos antes de aplicar modelos formales.

Recientemente, Python se ha consolidado como una de las herramientas más potentes para el análisis de datos gracias a su ecosistema de bibliotecas especializadas. En particular, pandas facilita la manipulación y limpieza de datos tabulares; matplotlib y seaborn permiten realizar visualizaciones gráficas efectivas que favorecen el entendimiento exploratorio; y otras herramientas como numpy ofrecen soporte computacional eficiente.

Esta guía introductoria al AED en Python presenta los conceptos y procedimientos fundamentales a través de un enfoque práctico y pedagógico. El análisis se desarrolla a partir de un conjunto de datos reales correspondientes a los resultados de las Pruebas Saber 11 del año 2020, disponibles en el portal oficial de datos abiertos del gobierno colombiano datos.gov.co. Este conjunto incluye información personal, socioeconómica y familiar de los estudiantes, así como sus puntajes por áreas, lo que permite ilustrar cómo el AED puede ser aplicado para explorar patrones educativos.

A lo largo del documento, se abordan temas como la carga e inspección inicial de los datos, el tratamiento de valores nulos, la generación de estadísticas descriptivas y la visualización de variables categóricas y numéricas. Todo el desarrollo se realiza en un entorno reproducible, con código comentado paso a paso y centrado en la lógica de exploración, lo que facilita su comprensión por parte de estudiantes e investigadores en formación.

Esta guía está dirigida a economistas, profesionales en formación y analistas interesados en introducirse al análisis de datos con Python, brindando herramientas esenciales para una lectura crítica y estructurada de los datos antes de cualquier modelado econométrico, inferencia causal o proyecto de Machine Learning.

Exploración inicial y limpieza

En esta sección aprenderemos sobre:

- **Librerías, carga y filtros**
- **Tratamiento de valores nulos y duplicados**
- **Exploración inicial de filas y columnas**
- **Estadísticas descriptivas para variables cuantitativas**
- **Frecuencia y valores únicos para variables cualitativas**

Librerías, carga y filtros

En esta subsección **importamos las librerías** de interés `pandas`, `matplotlib`, `seaborn` y `numpy` para la manipulación y visualización de datos. **Cargamos el dataset** en formato `.csv` con la función `read_csv` de `pandas` y **filtramos las columnas** de interés.

In [5]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
```

In [6]:
```
# Carga inicial del .csv
df  =  pd.read_csv(r"C:\Users\juanr\OneDrive\Escritorio\saber11_2020.csv")

# Creamos un vector con el nombre de las columnas/variables que nos interesan
columnas_interes = [
    "COLE_NATURALEZA",
    "ESTU_DEDICACIONLECTURADIARIA",
    "FAMI_ESTRATOVIVIENDA",
    "FAMI_EDUCACIONPADRE",
    "FAMI_TIENEINTERNET",
    "FAMI_TIENECOMPUTADOR",
    "FAMI_NUMLIBROS",
    "PUNT_LECTURA_CRITICA",
    "PUNT_MATEMATICAS",
    "PUNT_C_NATURALES",
    "PUNT_SOCIALES_CIUDADANAS",
    "PUNT_INGLES",
    "PUNT_GLOBAL",
]

# Filtramos por columnas de interés
data = df[columnas_interes]
data = data[
    (data['FAMI_ESTRATOVIVIENDA'] != 'Sin Estrato')
]
```

Tratamiento de valores nulos y duplicados

En esta subsección identificamos los **valores nulos y duplicados** mediante los métodos `isnull` y `duplicated` para después eliminarlos. **Es fundamental tener un dataset limpio y sin valores nulos para realizar una correcta visualización.**

In [8]:
```
# Identificar valores nulos

print(" Conteo de valores nulos por columna: \n")
data.isnull().sum()
```

```
Conteo de valores nulos por columna:
```

Out[8]:
```
COLE_NATURALEZA                    0
ESTU_DEDICACIONLECTURADIARIA     14361
FAMI_ESTRATOVIVIENDA             16952
FAMI_EDUCACIONPADRE              12954
FAMI_TIENEINTERNET               13569
FAMI_TIENECOMPUTADOR             19805
FAMI_NUMLIBROS                   13747
PUNT_LECTURA_CRITICA                 0
PUNT_MATEMATICAS                     0
PUNT_C_NATURALES                     0
PUNT_SOCIALES_CIUDADANAS             0
PUNT_INGLES                        229
PUNT_GLOBAL                          0
dtype: int64
```

In [9]:
```
# Identificar valores duplicados

print("Número de filas duplicadas: \n")
print(data.duplicated().sum())
```

```
Número de filas duplicadas:

132
```

In [10]:
```
# Eliminar filas con valores duplicados y nulos

def eliminar_nulos_duplicados(dataframe):
    dataframe = dataframe.drop_duplicates()
    dataframe = dataframe.dropna()
    print(f"Filas sin duplicados ni valores nulos: {dataframe.shape[0]}")
    return dataframe

data = eliminar_nulos_duplicados(data)
```

```
Filas sin duplicados ni valores nulos: 460287
```

Exploración inicial de filas y columnas

En esta subsección **realizamos una observación inicial de los valores de las filas y columnas del dataset**, así como a sus **dimensiones** y al **tipo de variables** que interpreta Python. Esto mediante funciones y atributos como `info`, `shape` y `head`.

In [12]:
```
# Información general del dataset

print("Información del dataset: \n")
data.info()

# Este códio imprime el nombre de las columnas del dataset, sus cantidad de valores
# "no nulos" y el tipo de variables.
# Donde:

# "object" hace referencia a una variable tipo texto (cualitativa)
# "int64" hace referencia a una variable numérica discreta (cuantitativa)
# "float64" hace referencia a una variable numércia continua (cuantitativa)
```

```
Información del dataset:

<class 'pandas.core.frame.DataFrame'>
Index: 460287 entries, 0 to 504871
Data columns (total 13 columns):
 #   Column                          Non-Null Count   Dtype
---  ------                          --------------   -----
 0   COLE_NATURALEZA                 460287 non-null  object
 1   ESTU_DEDICACIONLECTURADIARIA    460287 non-null  object
 2   FAMI_ESTRATOVIVIENDA            460287 non-null  object
 3   FAMI_EDUCACIONPADRE             460287 non-null  object
 4   FAMI_TIENEINTERNET              460287 non-null  object
 5   FAMI_TIENECOMPUTADOR            460287 non-null  object
 6   FAMI_NUMLIBROS                  460287 non-null  object
 7   PUNT_LECTURA_CRITICA            460287 non-null  int64
 8   PUNT_MATEMATICAS                460287 non-null  int64
 9   PUNT_C_NATURALES                460287 non-null  int64
 10  PUNT_SOCIALES_CIUDADANAS        460287 non-null  int64
 11  PUNT_INGLES                     460287 non-null  float64
 12  PUNT_GLOBAL                     460287 non-null  int64
dtypes: float64(1), int64(5), object(7)
memory usage: 49.2+ MB
```

In [13]:
```
# Dimensiones del dataset

print(" Dimensiones del dataset: \n")
print(f"Filas: {data.shape[0]}, Columnas: {data.shape[1]}")
```

```
 Dimensiones del dataset:

Filas: 460287, Columnas: 13
```

In [14]:
```
# Observación inicial de filas y columnas del dataset

print("Dataset filtrado con las columnas de interés: \n")
print(data.head())
```

```
Dataset filtrado con las columnas de interés:

  COLE_NATURALEZA  ESTU_DEDICACIONLECTURADIARIA  FAMI_ESTRATOVIVIENDA   \
0         OFICIAL            30 minutos o menos             Estrato 2
1         OFICIAL    No leo por entretenimiento             Estrato 3
2         OFICIAL         Entre 30 y 60 minutos             Estrato 1
4         OFICIAL             Entre 1 y 2 horas             Estrato 5
6      NO OFICIAL            30 minutos o menos             Estrato 3

                  FAMI_EDUCACIONPADRE  FAMI_TIENEINTERNET  FAMI_TIENECOMPUTADOR   \
0      Técnica o tecnológica completa                  Si                     Si
1  Secundaria (Bachillerato) completa                  Si                     No
2                 Primaria incompleta                  No                     No
4  Secundaria (Bachillerato) completa                  Si                     Si
6      Educación profesional completa                  Si                     Si

    FAMI_NUMLIBROS    PUNT_LECTURA_CRITICA   PUNT_MATEMATICAS   PUNT_C_NATURALES  \
0  26 A 100 LIBROS                      54                 65                 41
1    0 A 10 LIBROS                      57                 43                 46
2    0 A 10 LIBROS                      59                 72                 63
4    0 A 10 LIBROS                      37                 48                 44
6  26 A 100 LIBROS                      49                 68                 50

   PUNT_SOCIALES_CIUDADANAS   PUNT_INGLES   PUNT_GLOBAL
0                        33          55.0           244
1                        49          33.0           238
2                        68          59.0           325
4                        32          43.0           202
6                        51          65.0           277
```

In [15]:
```
# También el dataset se puede visualizar en formato datafream

data.head()
```

Out[15]:

	COLE_NATURALEZA	ESTU_DEDICACIONLECTURADIARIA	FAMI_ESTRATOVIVIENDA	FAMI_
0	OFICIAL	30 minutos o menos	Estrato 2	Té
1	OFICIAL	No leo por entretenimiento	Estrato 3	Secu
2	OFICIAL	Entre 30 y 60 minutos	Estrato 1	
4	OFICIAL	Entre 1 y 2 horas	Estrato 5	Secu
6	NO OFICIAL	30 minutos o menos	Estrato 3	Ed

Estadísticas descriptivas para variables cuantitativas

Esta subsección nos permite conocer de manera rápida y sencilla las principales **estadísticas descriptivas de las variables cuantitativas**, aquellas tipo `int64` y `float64`, mediante el método `describe`.

In [17]:
```
print("Estadísticas descriptivas para variables numéricas: \n")

numeric_columns = data.select_dtypes(include=['float64', 'int64'])
numeric_columns.describe()
```

```
Estadísticas descriptivas para variables numéricas:
```

Out[17]:

	PUNT_LECTURA_CRITICA	PUNT_MATEMATICAS	PUNT_C_NATURALES	PUNT_SOCIALE
count	460287.000000	460287.000000	460287.000000	
mean	52.545892	51.476103	48.650362	
std	9.999261	11.514633	10.369718	
min	0.000000	0.000000	0.000000	
25%	45.000000	43.000000	41.000000	
50%	53.000000	51.000000	48.000000	
75%	60.000000	59.000000	56.000000	
max	100.000000	100.000000	100.000000	

Frecuencia y valores únicos para variables cualitativas

El código de esta subsección nos permite identificar la **frecuencia y valores únicos que toman las variables cualitativas**, aquellas tipo `object`. Esto mediante métodos como `value_counts`.

In [19]:
```
print("Frecuencia y valores únicos por variable cualitativa: \n")
object_columns = data.select_dtypes(include=['object'])

for col in object_columns.columns:
    print(f"\nVariable: {col}")
    print(data[col].value_counts())
```

Frecuencia y valores únicos por variable cualitativa:

Variable: COLE_NATURALEZA
COLE_NATURALEZA
OFICIAL 354624
NO OFICIAL 105663
Name: count, dtype: int64

Variable: ESTU_DEDICACIONLECTURADIARIA
ESTU_DEDICACIONLECTURADIARIA
30 minutos o menos 177515
Entre 30 y 60 minutos 124807
No leo por entretenimiento 90035
Entre 1 y 2 horas 47674
Más de 2 horas 20256
Name: count, dtype: int64

Variable: FAMI_ESTRATOVIVIENDA
FAMI_ESTRATOVIVIENDA
Estrato 2 177969
Estrato 1 148683
Estrato 3 101097
Estrato 4 22836
Estrato 5 6761
Estrato 6 2941
Name: count, dtype: int64

Variable: FAMI_EDUCACIONPADRE
FAMI_EDUCACIONPADRE
Secundaria (Bachillerato) completa 113722
Primaria incompleta 84631
Secundaria (Bachillerato) incompleta 61118
Educación profesional completa 44682
Primaria completa 42631
Técnica o tecnológica completa 31860
No sabe 27978
Ninguno 17842
Educación profesional incompleta 9727
Técnica o tecnológica incompleta 9623
Postgrado 9194
No Aplica 7279
Name: count, dtype: int64

Variable: FAMI_TIENEINTERNET
FAMI_TIENEINTERNET
Si 328043
No 132244
Name: count, dtype: int64

Variable: FAMI_TIENECOMPUTADOR
FAMI_TIENECOMPUTADOR
Si 284401
No 175886
Name: count, dtype: int64

```
Variable: FAMI_NUMLIBROS
FAMI_NUMLIBROS
0 A 10 LIBROS         190791
11 A 25 LIBROS        143460
26 A 100 LIBROS        96207
MÁS DE 100 LIBROS      29829
Name: count, dtype: int64
```

Visualización de datos

En esta sección aprenderemos sobre:

1. Visualización individual de variables cuantitativas

2. Visualización conjunta de variables cuantitativas

3. Visualización individual de variables cualitativas

4. Visualización conjunta de variables cualitativas

5. Visualización conjunta de variables cuantitativas y cualitativas

Visualización individual de variables cuantitativas

En esta subsección introduciremos los **conceptos y tipos de gráficos más comunes para la visualización individual de una variable cuantitativa.**

Aprenderemos sobre:

1. Histograma

2. Diagrama de caja y bigotes (Box-Plot)

3. Gráfico de violín (Violin-Plot)

4. Gráfico de densidad (Kernel Density-Plot)

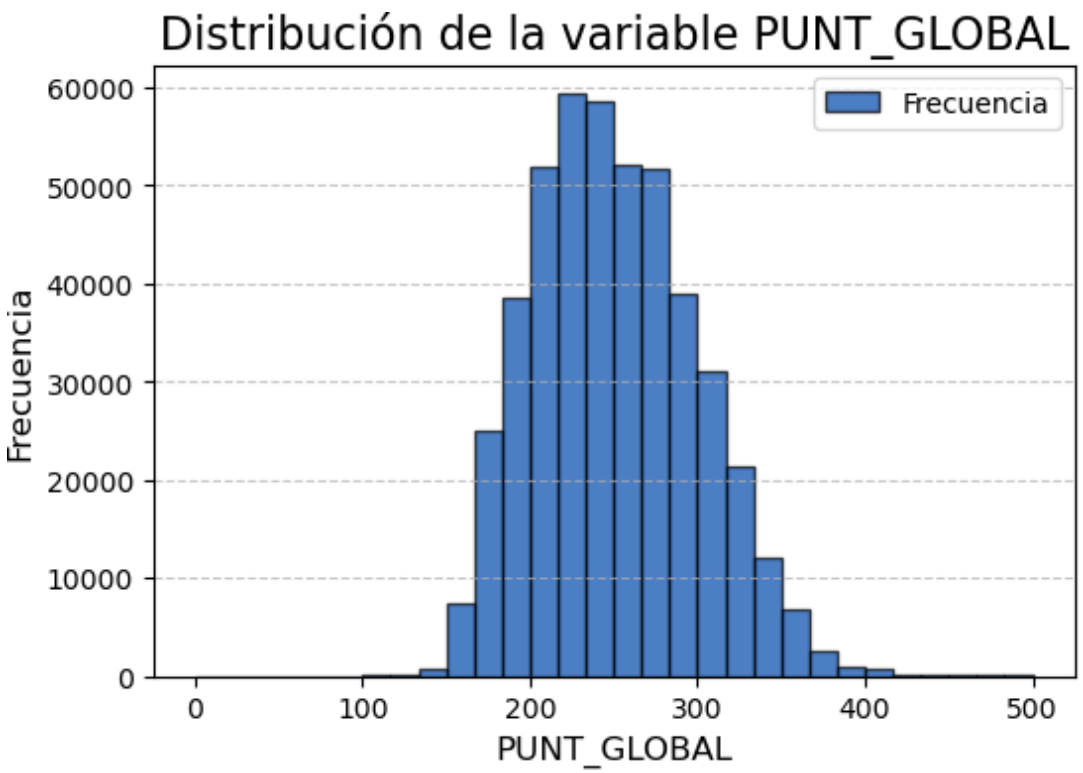
In [22]:
```
# Creamos vectores que almacenan variables cuantitativas específicas y de interés p

quantitative_vars = data[['PUNT_GLOBAL', 'PUNT_MATEMATICAS', 'PUNT_C_NATURALES', 'P
quantitative_vars_1 = data[['PUNT_GLOBAL', 'PUNT_MATEMATICAS', 'PUNT_C_NATURALES']]
quantitative_vars_2 = data[['PUNT_INGLES', 'PUNT_SOCIALES_CIUDADANAS', 'PUNT_LECTUR
```

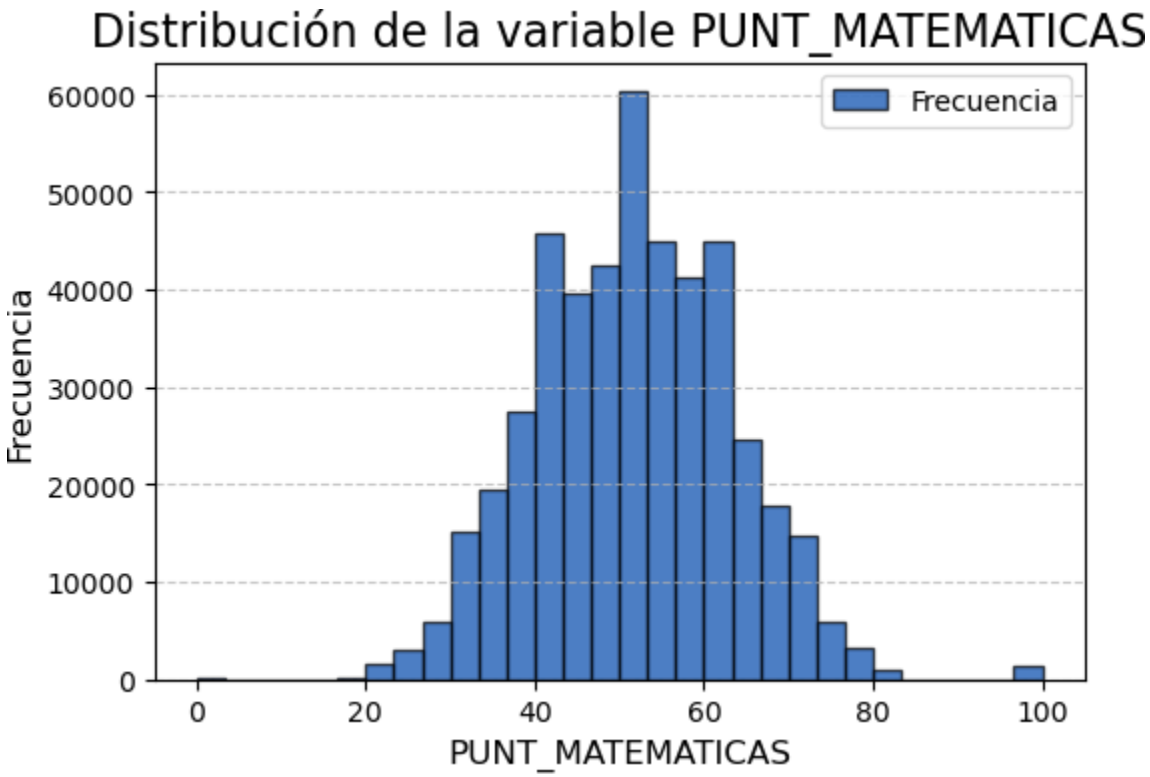
Histograma

Un histograma es un gráfico de barras que muestra la distribución de un conjunto de datos en rangos o segmentos. La altura de cada barra representa la frecuencia con la que

aparece la variable en cada rango. Es ideal para visualizar la forma de la distribución, la centralidad y la dispersión.

In [24]:
```
for var in quantitative_vars_1:
    plt.figure(figsize=(6, 4))

    # Histograma
    plt.hist(data[var], bins=30, alpha=0.7, color="#0047AB", edgecolor='black',
             label='Frecuencia')
    # Configuración del gráfico
    plt.title(f'Distribución de la variable {var}', fontsize=16)
    plt.xlabel(var, fontsize=12)
    plt.ylabel('Frecuencia', fontsize=12)
    plt.legend(loc='best')
    plt.grid(axis='y', linestyle='--', alpha=0.7)
    plt.show()
```





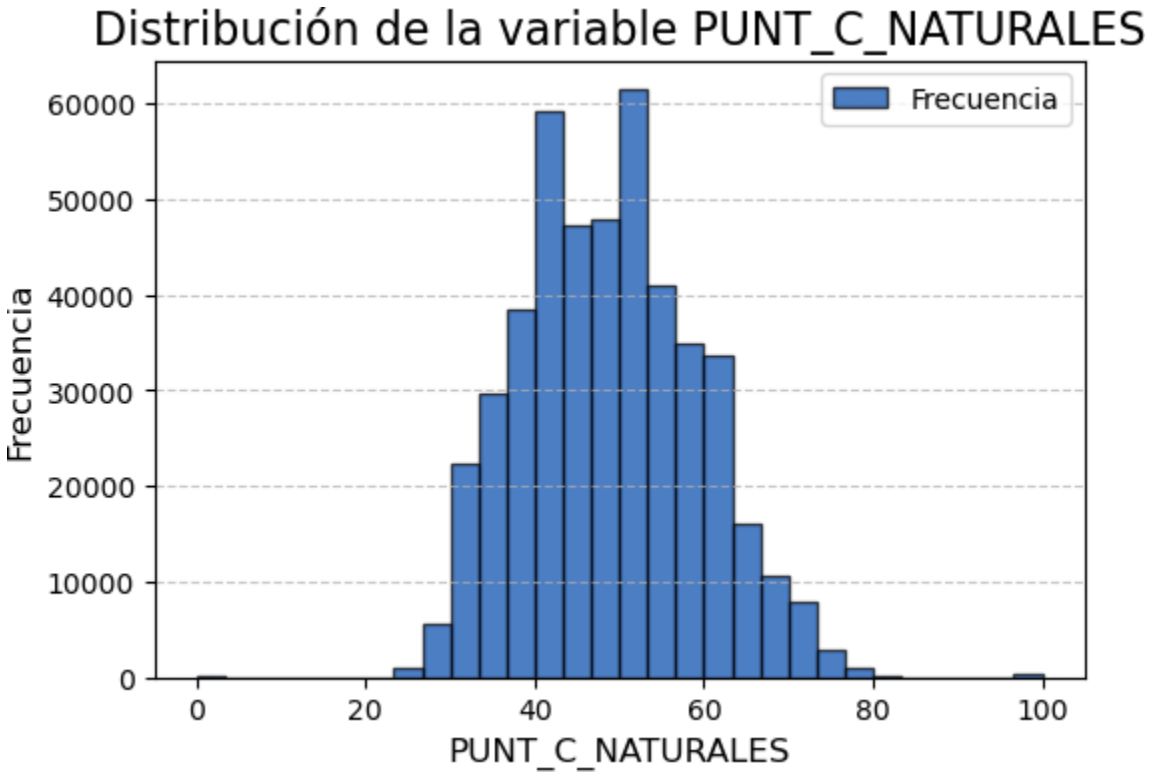


Diagrama de caja y bigotes (Box-Plot)

Un Box-Plot muestra la distribución de un conjunto de datos a través de sus cuartiles, mínimos, máximos y datos atípicos. Permite identificar valores atípicos y comparar distribuciones.

1. Los límites de la caja son el Q_1 (25%) y el Q_3 (75%).

2. La línea dentro de la caja es el dato de la mitad, mediana o $Q2$ (50%).

3. Los límites inferior y superior de los bigotes son:

- Límite Inferior = $Q_1 - 1.5 \cdot RIC$
- Límite Superior = $Q_3 + 1.5 \cdot RIC$

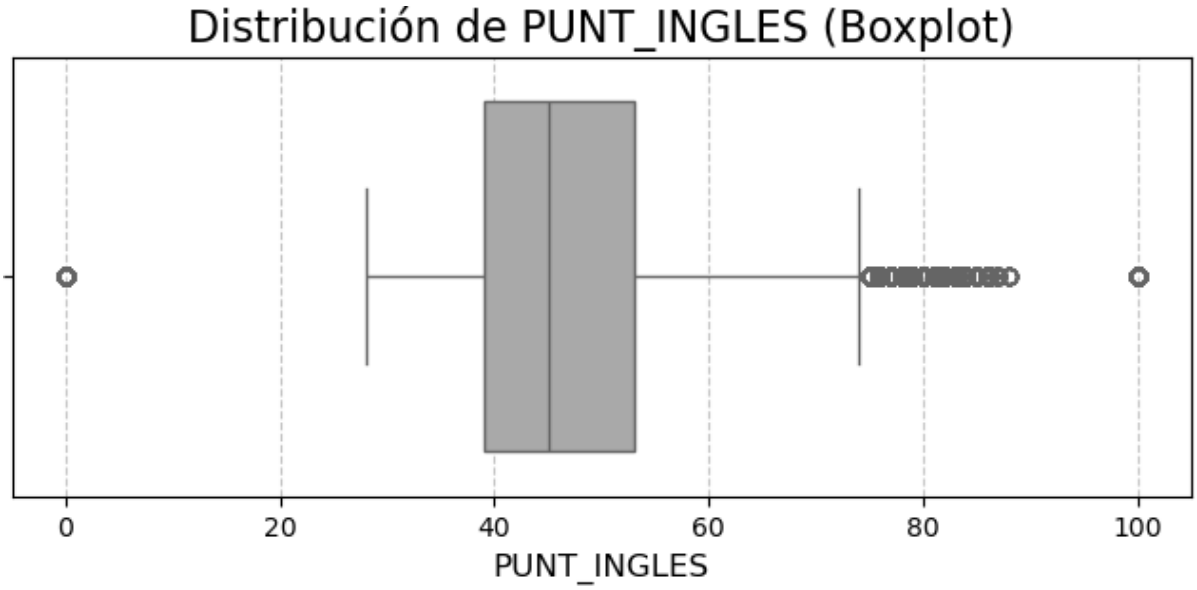
Donde RIC se define como **Rango Interquartílico** y se calcula como la diferencia entre el Q_3 y Q_1:

- $RIC = Q_3 - Q_1$

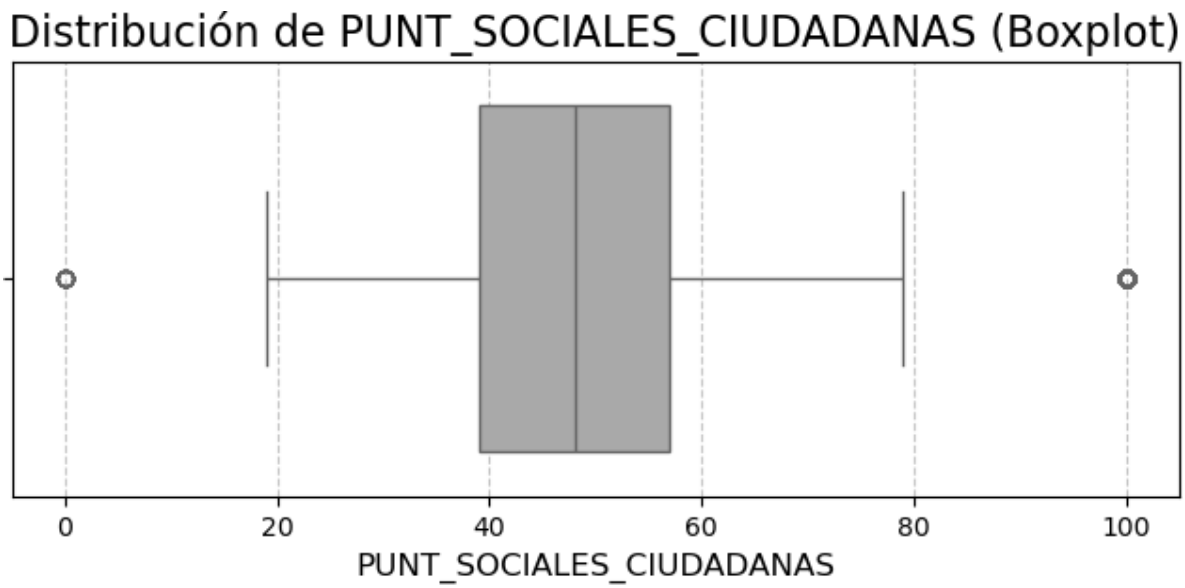
Aquellos datos que se encuentran por fueral del Límite Inferior\Superior son considerados datos atípicos

In [26]:
```
for var in quantitative_vars_2:
    plt.figure(figsize=(6, 4))

# Boxplot
    plt.figure(figsize=(8, 3))
    sns.boxplot(x=data[var], color="#A9A9A9")
    plt.title(f'Distribución de {var} (Boxplot)', fontsize=16)
    plt.xlabel(var, fontsize=12)
    plt.grid(axis='x', linestyle='--', alpha=0.7)
    plt.show()
```

```
<Figure size 600x400 with 0 Axes>
```




```
<Figure size 600x400 with 0 Axes>
```




```
<Figure size 600x400 with 0 Axes>
```

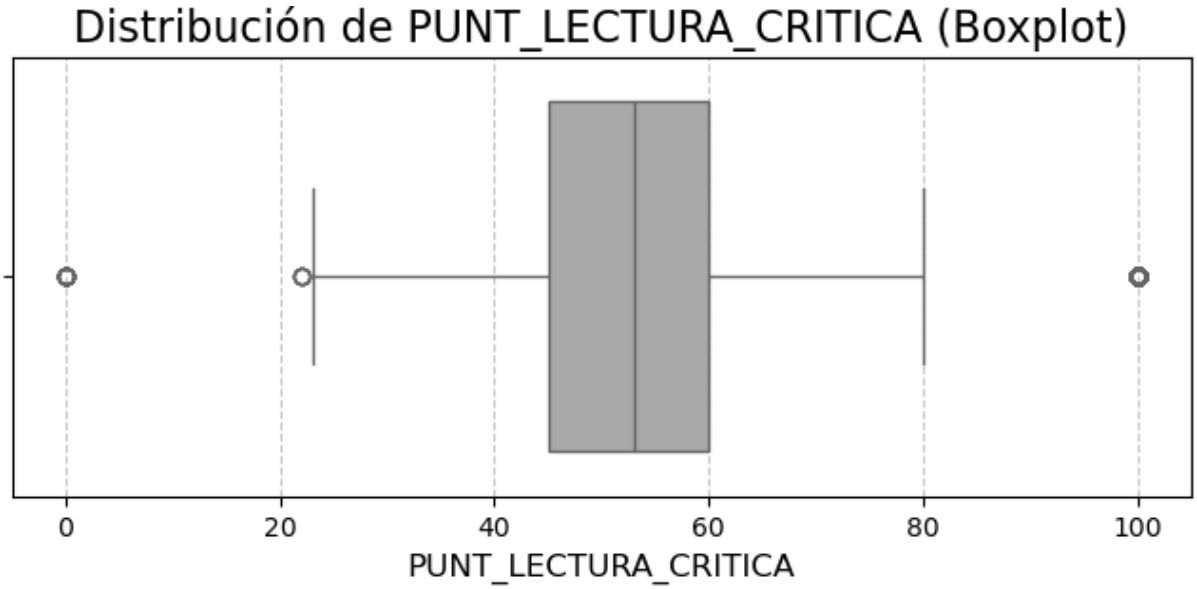


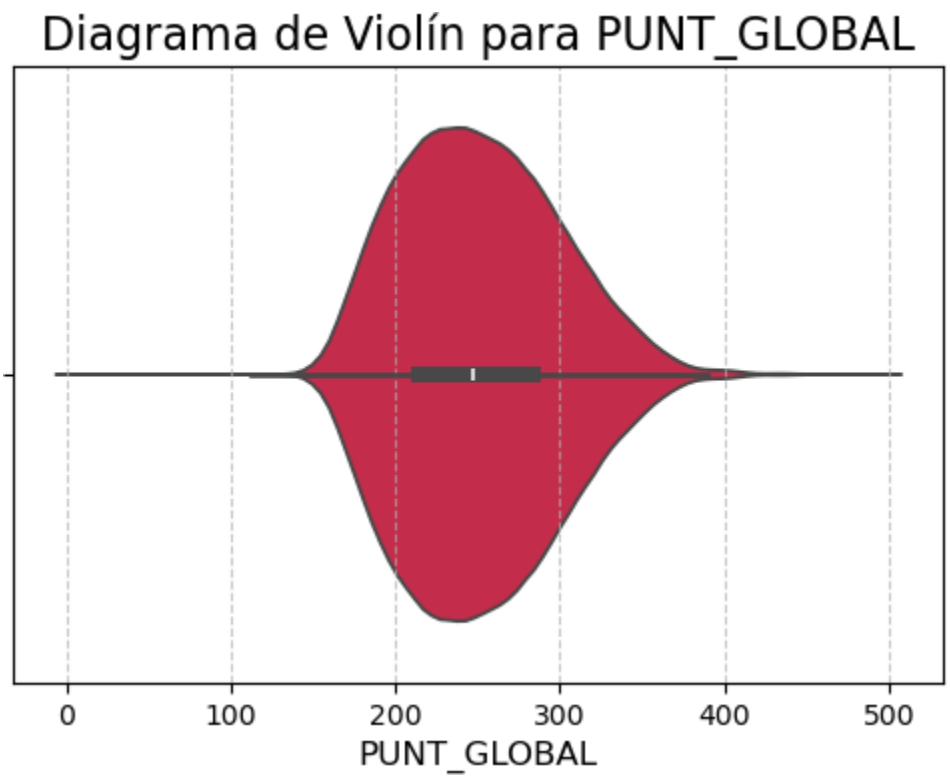
Gráfico de violín (Violin-Plot)

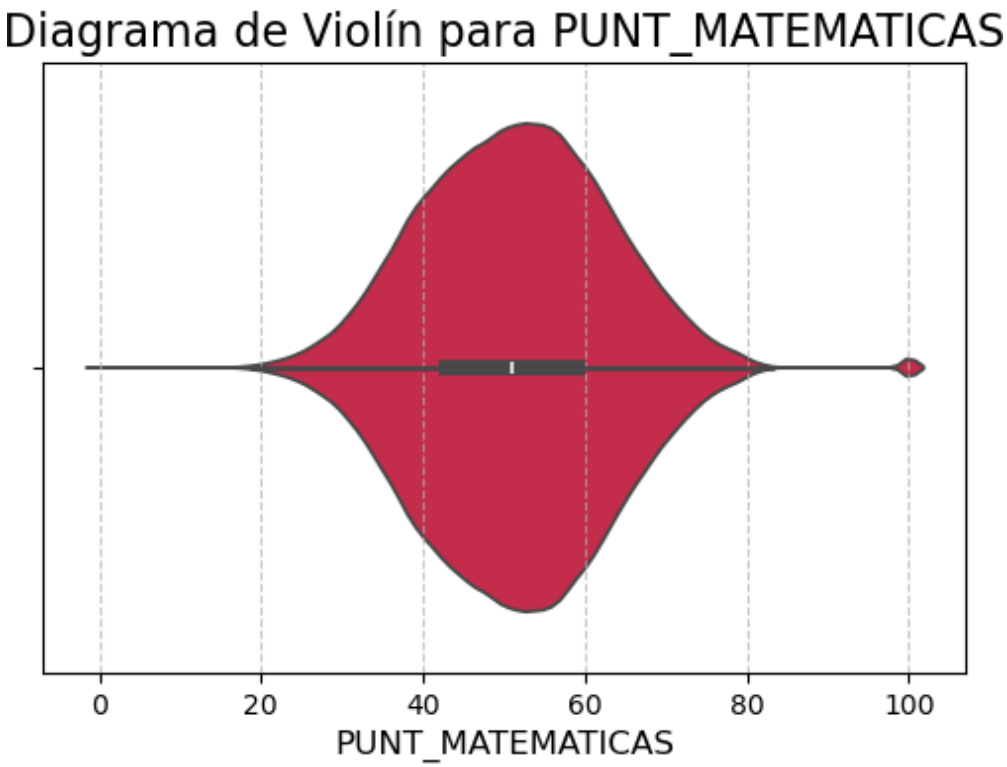
Un gráfico de violín combina las herramientas del Box-Plot y del Gráfico de densidad. En su interior, contiene:

- Dos rectángulos pequeños que reflejan el Q_1 y Q_3.
- En el espacio entre estos dos rectángulos se encuentra la mediana Q_2.
- La alutra/ancho del violín representa la frecuencia/densidad de datos en torno a su valor en el eje x.

In [28]:
```
for var in quantitative_vars_1:
    plt.figure(figsize=(6, 4))
    sns.violinplot(x=data[var],  color="#DC143C")
    plt.title(f'Diagrama de Violín para {var}', fontsize=16)
    plt.xlabel(var, fontsize=12)
```

```
    plt.grid(axis='x', linestyle='--', alpha=0.7)
    plt.show()
```





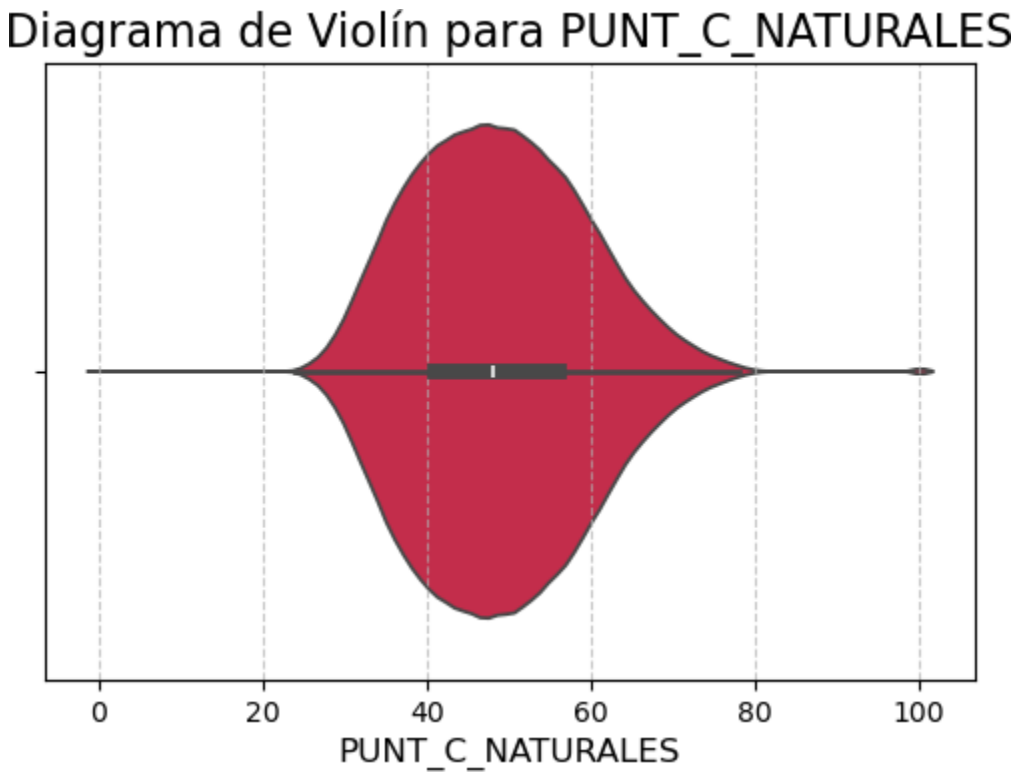
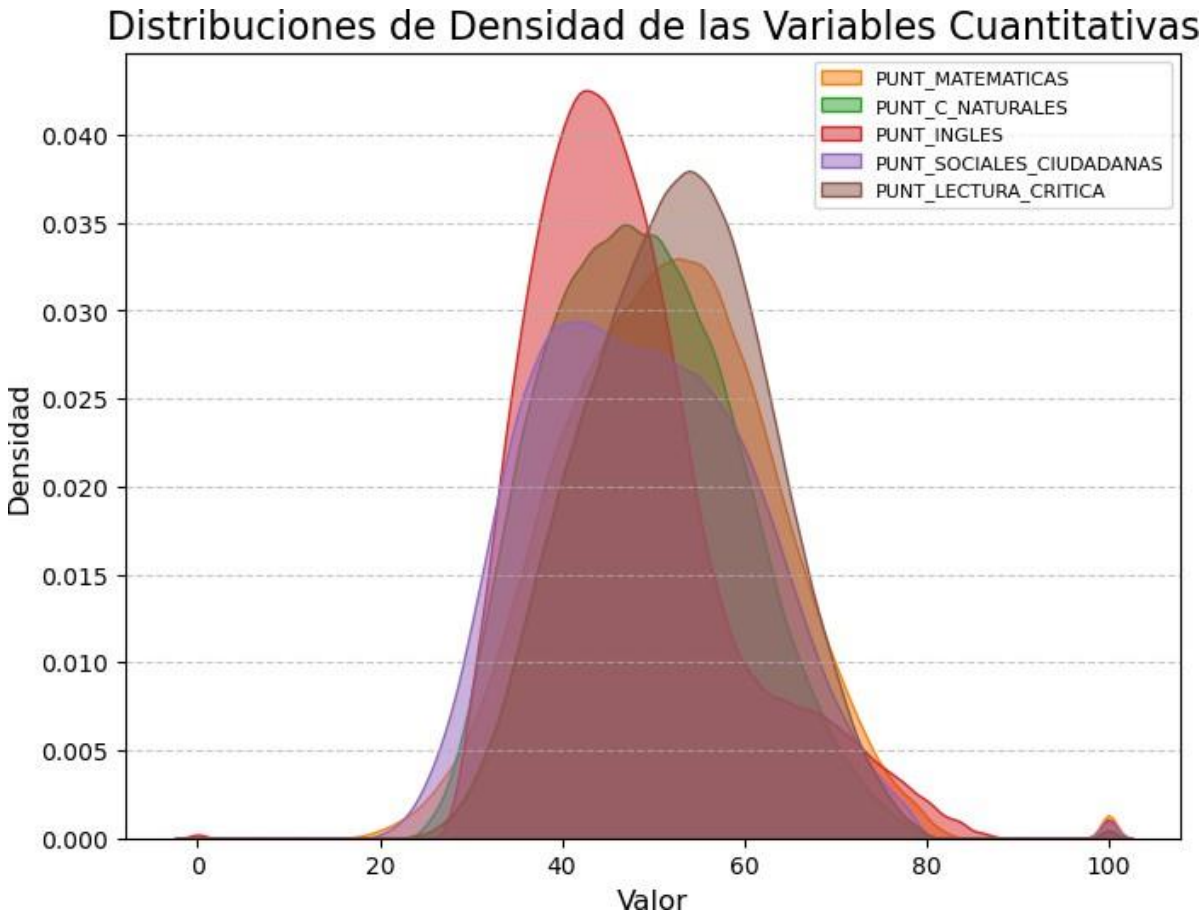


Gráfico de densidad (Kernel Density-Plot)

Un **gráfico de densidad suaviza la distribución de las variables cuantitaivas** representadas inicalmente en el histograma. A pesar de que se pierde información detallada de cada variable, **es útil para hacer análisis entre variables que presenten una misma unidad de medida y similar rango.**

In [30]:
```
# Gráfico de densidad combinado para todas las variables cuantitativas (sin incluir
plt.figure(figsize=(8, 6))

# Paleta de colores para identificar las variables fácilmente
colors = sns.color_palette('tab10', len(quantitative_vars))

for i, var in enumerate(quantitative_vars):
    if var != 'PUNT_GLOBAL':
        sns.kdeplot(data[var], fill=True, alpha=0.5, label=var, color=colors[i])

# Configuración del gráfico
plt.title('Distribuciones de Densidad de las Variables Cuantitativas', fontsize=16)
plt.xlabel('Valor', fontsize=12)
plt.ylabel('Densidad', fontsize=12)
plt.legend(loc='best', fontsize=8)  # Tamaño más pequeño para la leyenda
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```



Visualización conjunta de variables cuantitativas

En esta subsección introduciremos los **conceptos y tipos de gráficos más comunes para la visualización conjunta variables cuantitativas.**

Aprenderemos sobre:

1. Gráfico de dispersión (Scatter-Plot)

2. Matriz de correlación (Heatmap)

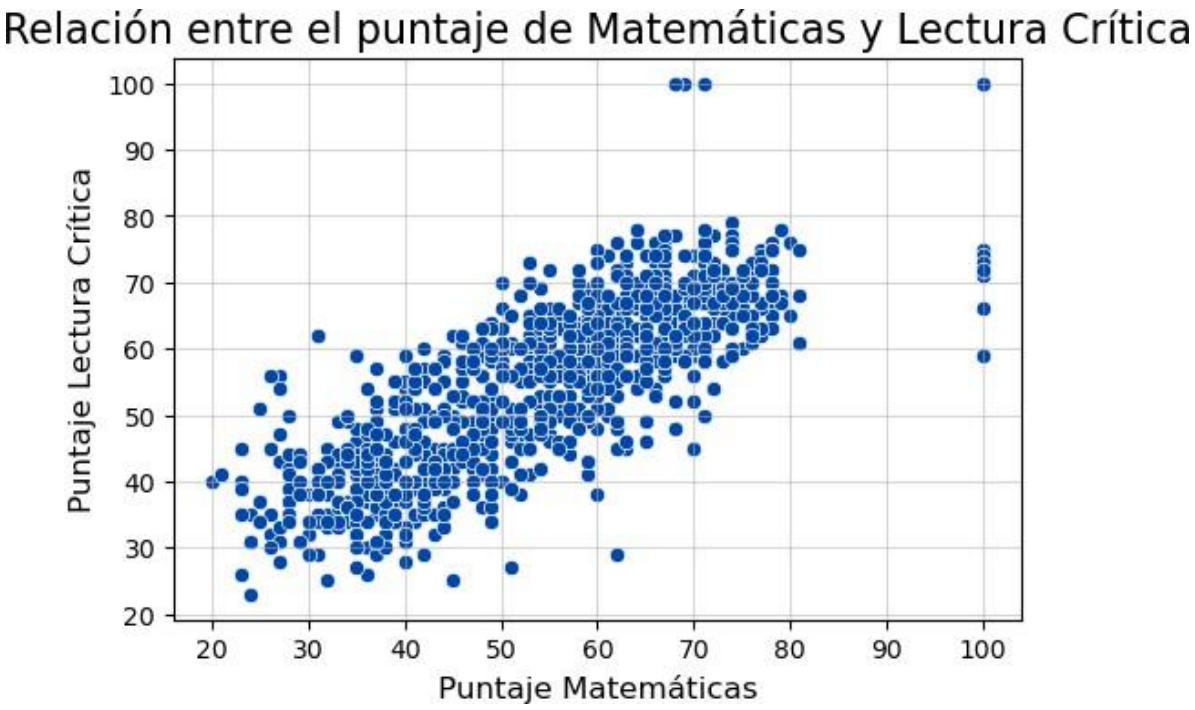
3. Gráfico de pares (Pair-Plot)

In [32]:
```
# Para los gráficos de dispersion rreamos una muestra aleatoria de 1000 observacion
# estudiantes pertenecientes al estrato 5. Esto es útil para los gráficos de disper
# se recomienda para una gran cantidad de observaciones.

f_data = data[data['FAMI_ESTRATOVIVIENDA'].isin(['Estrato 5'])]
f_data_sample = f_data.sample(n=1000)
```

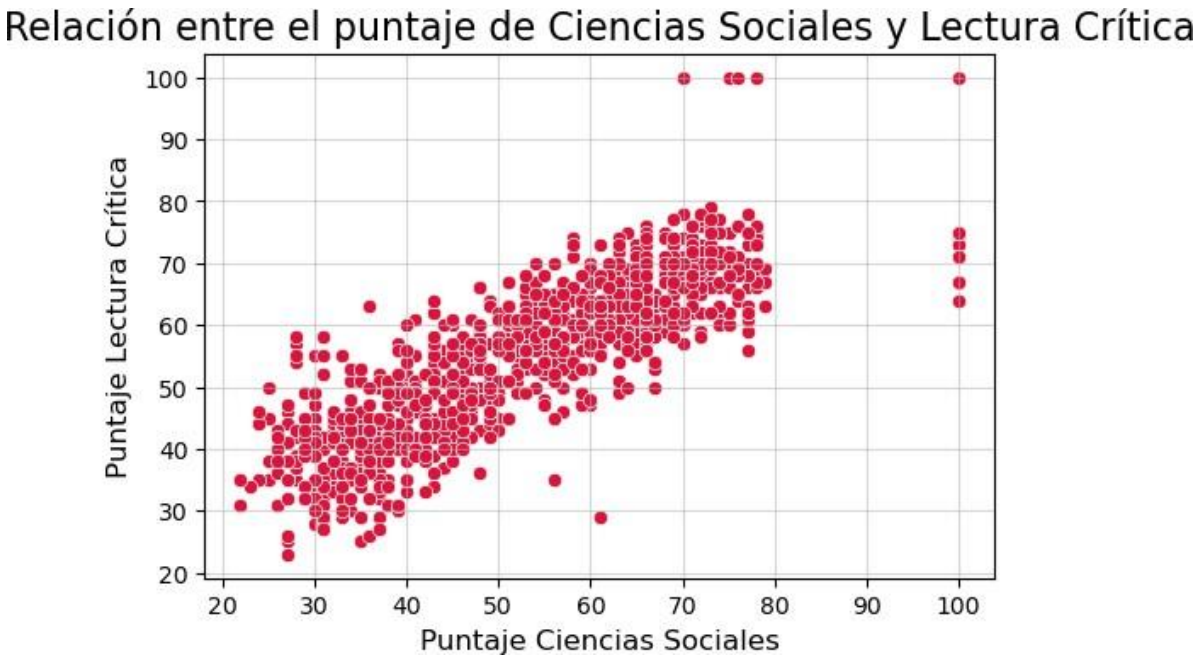
Gráfico de dispersión (Scatter-Plot)

Un Gráfico de dispersión o Scatter-Plot muestra la ubicación de una observación en un plano mediante los valores que toma para las dos variables cuantitativas de interés

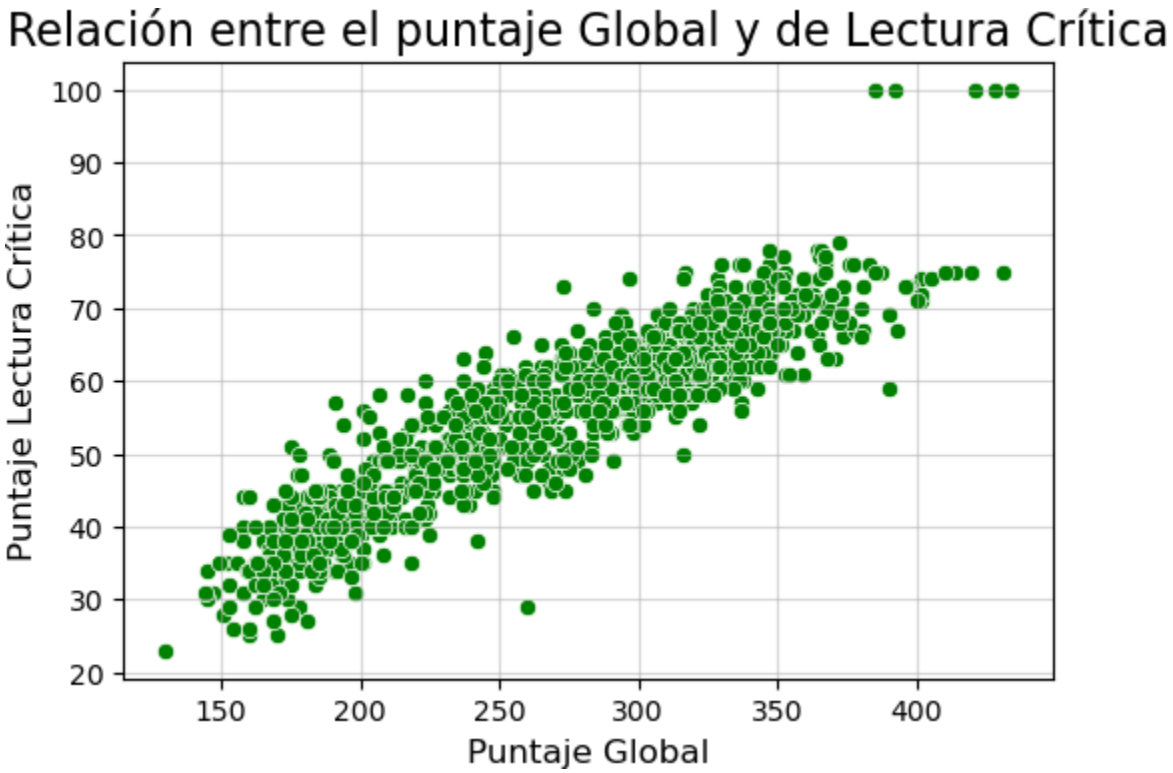
In [34]:
```
#Scatter-Plot
plt.figure(figsize = (6,4))
sns.scatterplot(x=f_data_sample['PUNT_MATEMATICAS'],   y=f_data_sample['PUNT_LECTURA_
plt.title('Relación entre el puntaje de Matemáticas y Lectura Crítica', fontsize=16
plt.xlabel('Puntaje Matemáticas', fontsize=12)
plt.ylabel('Puntaje Lectura Crítica', fontsize=12)
plt.grid(alpha=0.5)
plt.show()
```



In [35]:
```
plt.figure(figsize = (6,4))
sns.scatterplot(x=f_data_sample['PUNT_SOCIALES_CIUDADANAS'],   y=f_data_sample['PUNT_
plt.title('Relación entre el puntaje de Ciencias Sociales y Lectura Crítica', fonts
plt.xlabel('Puntaje Ciencias Sociales', fontsize=12)
plt.ylabel('Puntaje Lectura Crítica', fontsize=12)
plt.grid(alpha=0.5)
plt.show()
```



In [36]:
```
plt.figure(figsize = (6,4))
sns.scatterplot(x=f_data_sample['PUNT_GLOBAL'], y=f_data_sample['PUNT_LECTURA_CRITI
plt.title('Relación entre el puntaje Global y de Lectura Crítica', fontsize=16)
plt.xlabel('Puntaje Global', fontsize=12)
plt.ylabel('Puntaje Lectura Crítica', fontsize=12)
plt.grid(alpha=0.5)
plt.show()
```



Matriz de correlación (Heatmap)

Una matriz de correlación es una tabla que muestra los coeficientes de correlación entre todas las variables numéricas de un conjunto de datos. Cada celda de la tabla representa el coeficiente de correlación entre dos variables específicas. Los coeficientes de correlación varían entre -1 y 1, donde:

- Si el valor es cercano a +1, se acepta una fuerte correlación positiva
- Si el valor es cercano a -1, se acepta una fuerte Correlación negativa
- Si el valor es cercano a 0, se acepta una débil o nula correlación

La fórmula del coeficiente de correlación es:

$$corr(x, y) = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Donde:

x_i : El valor individual de la variable x.
y_i : El valor individual de la variable y.
$\bar{x}$: La media aritmética de la variable x.
$\bar{y}$: La media aritmética de la variable y.
n : El número total de observaciones.
$\sum_{i=1}^{n}$: La suma de los valores desde $i = 1$ hasta $i = n$.
$\sqrt{}$: La raíz cuadrada.

In [38]:
```
# Seleccionar únicamente variables de tipo float64 e int64
f_numeric_data = f_data.select_dtypes(include=['float64', 'int64'])

plt.figure(figsize=(12, 10))  # Tamaño más amplio
corr_matrix = f_numeric_data.corr()
sns.heatmap(
    corr_matrix,
    annot=True,
    fmt='.2f',
    cmap='Reds',
    square=True,
    linewidths=0.5,  # Líneas entre celdas
    cbar_kws={'shrink': 0.8, 'aspect': 40},  # Barra de color ajustada
    annot_kws={'size': 10, 'color': 'black'}  # Sin negrilla en las anotaciones
)
plt.title('Matriz de Correlación para Variables Numéricas', fontsize=16, color='dar
plt.xticks(fontsize=12, rotation=45, ha='right')  # Rotación y estilo de etiquetas
plt.yticks(fontsize=12)  # Estilo de etiquetas del eje y
plt.show()
```

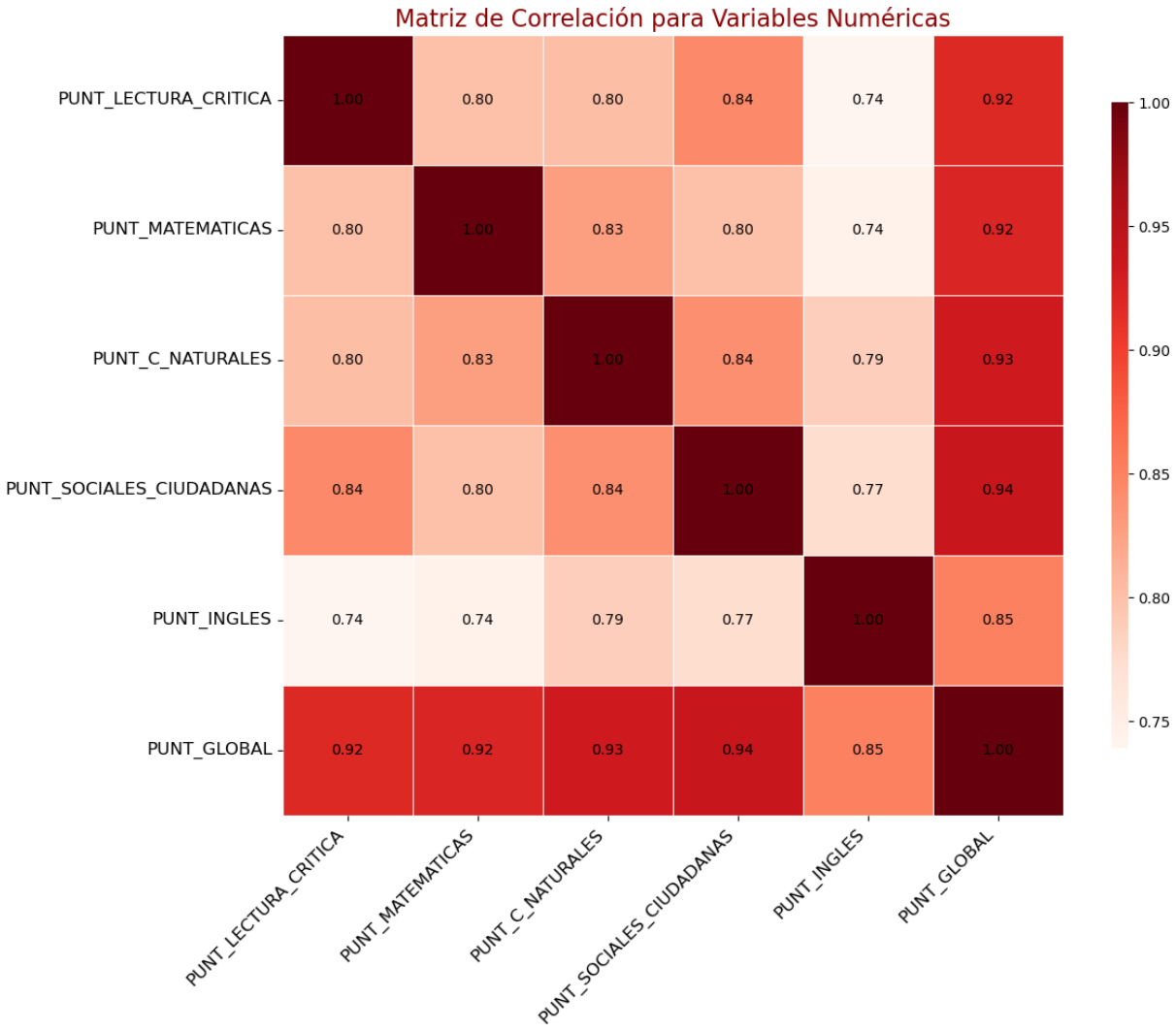
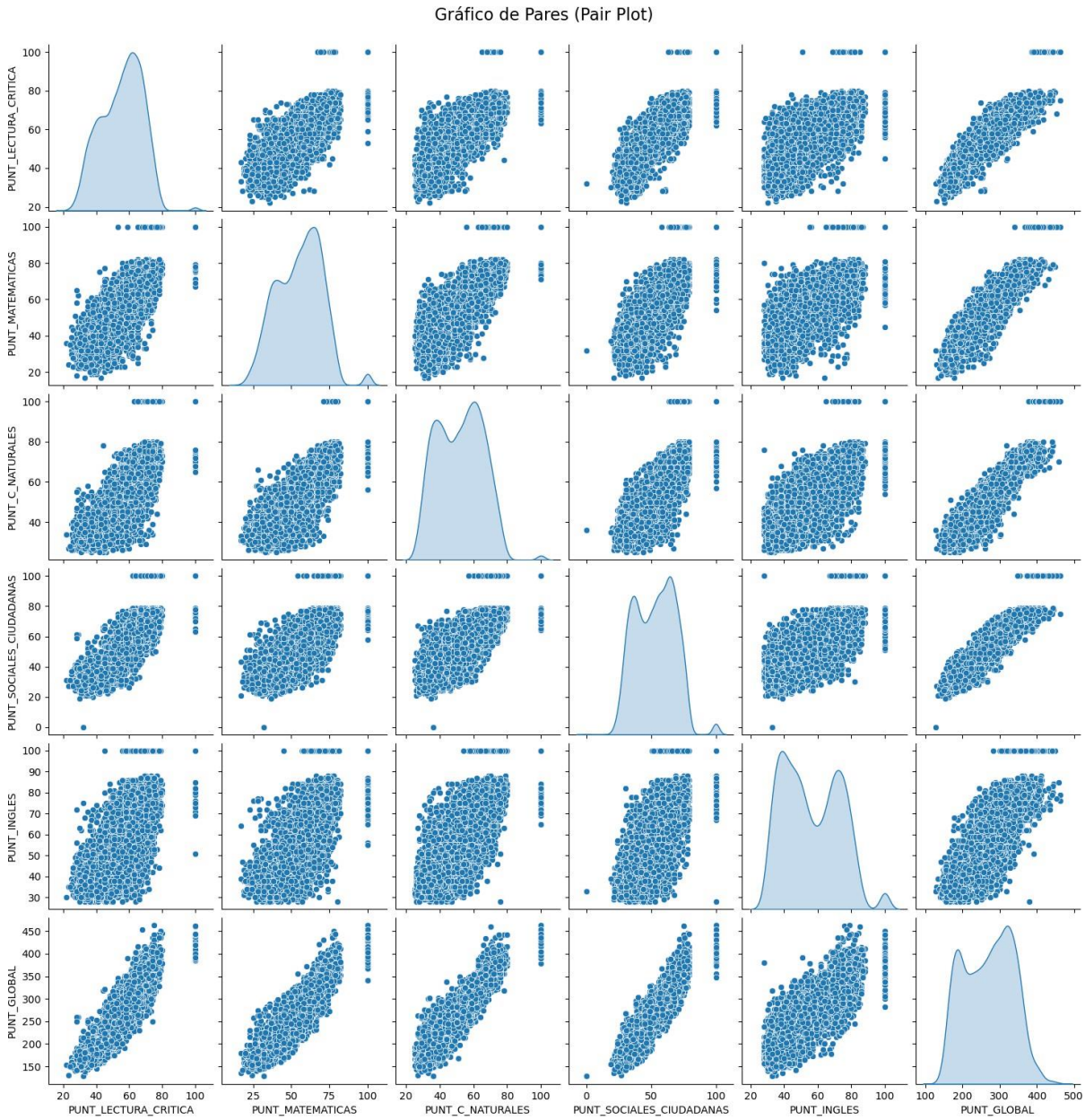


Gráfico de pares (Pair-Plot)

Un pairplot te permite visualizar de manera rápida y general las relaciones entre múltiples variables numéricas. Combina **gráficos de dispersión (scatter plots)** y **gráficos de densidad (KDE plots)** para mostrar relaciones entre múltiples variables cuantitativas.

- La diagonal de la matriz muestra la distribución univariada para cada variable.
- Los demás elementos de la matriz exponen los gráficos de dispersión para las variables de su respectiva fila y columna.
- La matriz se podría llegar a considerar en cierto sentido simétrica, pues las variables de sus ejes son las mismas.

In [40]:
```
sns.pairplot(data=f_numeric_data, diag_kind='kde', corner=False)
plt.suptitle('Gráfico de Pares (Pair Plot)', y=1.02, fontsize=16)
plt.show()

# Este gráfico sí se realizó con los datos poblaciones de los estudiantes de vivien
# No con una muestra de esta población.
```



Visualización individual de variables cualitativas

En esta subsección introduciremos los **conceptos y tipos de gráficos más comunes para la visualización individual de una variable cualitativa.**

Aprenderemos sobre:

1. Gráfico de barras (Bar-Plot)

2. Gráfico de pastel (Pie-Chart)

3. Gráfico de pares (Donut-Chart)

Gráfico de barras (Bar-Plot)

Un gráfico de barras o Bar-Plot muestra la frecuencia de las distintas categorías que toma una variable cualitativa.

In [43]:
```
sns.countplot(x=data['FAMI_ESTRATOVIVIENDA'], palette='deep',  hue=data['FAMI_ESTRA
plt.title('Distribución de Estratos de Vivienda', fontsize=16)
plt.xlabel('Estrato de Vivienda', fontsize=12)
plt.ylabel('Frecuencia', fontsize=12)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.show()
```

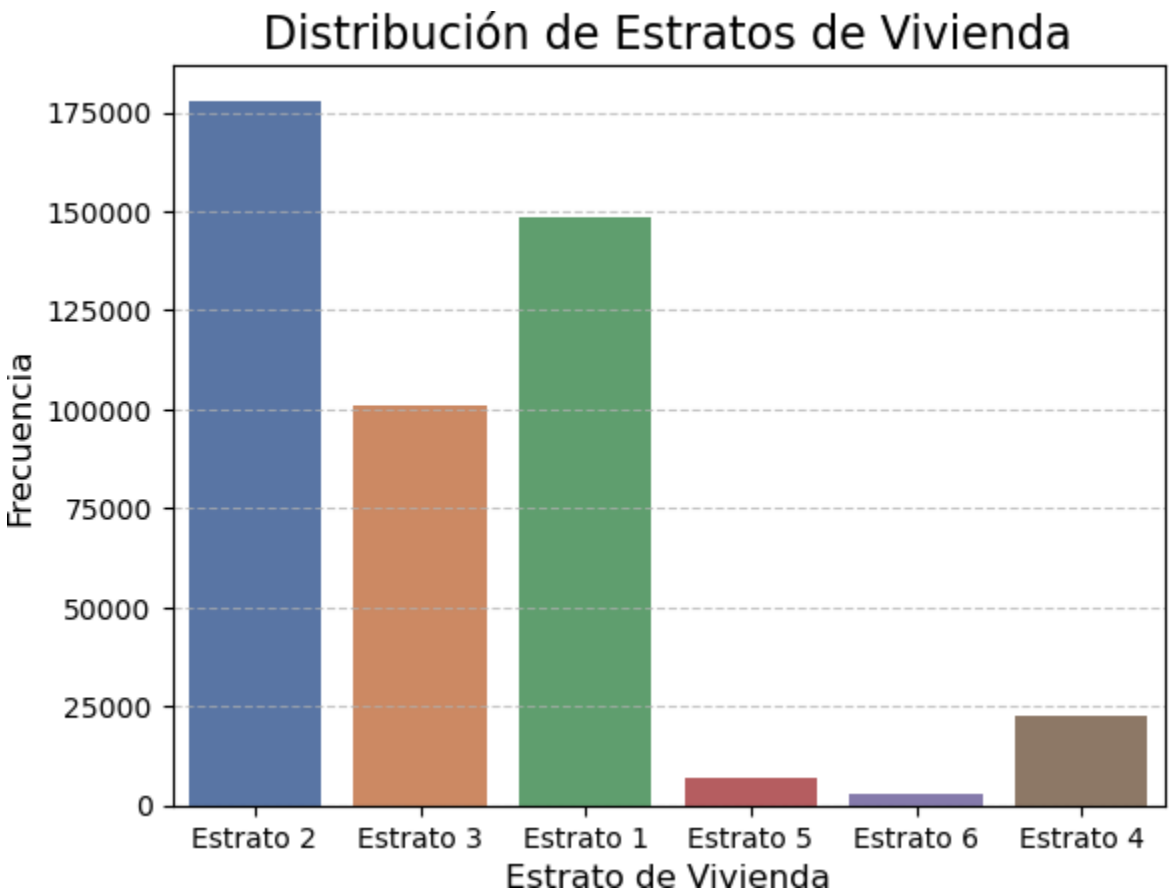


Gráfico de pastel (Pie-chart)

Un gráfico de pastel o Pie-Chart muestra la proporción de cada categoría como un sector del círculo.

In [45]:
```
data['FAMI_NUMLIBROS'].value_counts().plot.pie(
    autopct='%1.1f%%',  startangle=90,  colors=sns.color_palette("BrBG"))
plt.title('Proporción del rango de número de libros en las casas de los estudiantes
plt.ylabel('')
plt.show()
```

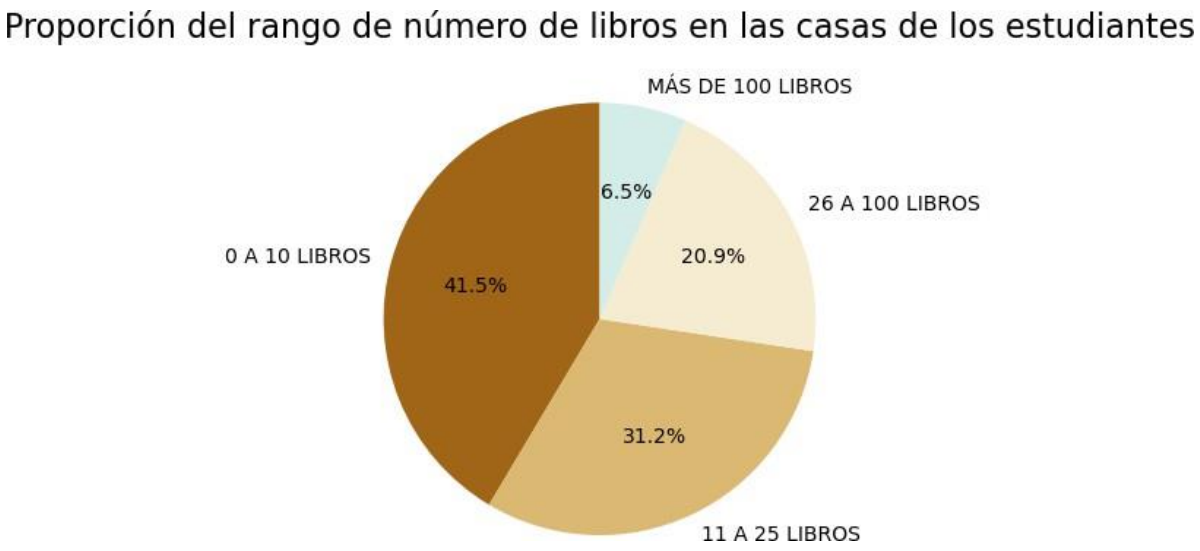


Gráfico de dona (Donut-Chart)

Similar al gráfico de pastel, pero con un círculo vacío en medio.

In [47]:
```
counts = data['ESTU_DEDICACIONLECTURADIARIA'].value_counts()
plt.pie(counts, labels=counts.index, autopct='%1.1f%%', startangle=90, colors=sns.c
plt.title('Proporción del tiempo de lectura diaria de los estudiantes', fontsize=16
plt.show()
```

Tabla de frecuencia absoluta y relativa

Una tabla de frecuencia absoluta y relativa muestra estas medidas para las categorías de una variable cualitativa.

In [49]:
```
freq_table = data['FAMI_ESTRATOVIVIENDA'].value_counts().reset_index()
freq_table.columns = ['Categoría', 'Frecuencia']
freq_table['Proporción (%)'] = round( freq_table['Frecuencia'] / freq_table['Frecue

freq_table
```

Out[49]:

	Categoría	Frecuencia	Proporción (%)
0	Estrato 2	177969	38.66
1	Estrato 1	148683	32.30
2	Estrato 3	101097	21.96
3	Estrato 4	22836	4.96
4	Estrato 5	6761	1.47
5	Estrato 6	2941	0.64

Visualización conjunta de variables cualitativas

En esta subsección introduciremos los **conceptos y tipos de gráficos más comunes para la visualización conjunta de variables cualitativas.**

Aprenderemos sobre:

1. Tablas dinámicas

2. Gráfico de barras apilado

3. Gráfico de barras agrupado

Tablas dinámicas

Las tablas dinámicas son herramientas esenciales para osbervar la frecuencia relativa y/o absoluta (y cualquier otra medida) entre la intersección de las distintas categorías de dos variables cualitativas.

In [52]:
```
cross_tab_totales = pd.crosstab(
    data['FAMI_TIENEINTERNET'],
    data['FAMI_ESTRATOVIVIENDA'],
    margins=True,
    margins_name="Total"
)

cross_tab_totales
```

Out[52]:

FAMI_ESTRATOVIVIENDA	Estrato 1	Estrato 2	Estrato 3	Estrato 4	Estrato 5	Estrato 6	Total
FAMI_TIENEINTERNET							
No	75871	41334	10847	2404	1072	716	132244
Si	72812	136635	90250	20432	5689	2225	328043
Total	148683	177969	101097	22836	6761	2941	460287

También se pueden visualizar las participaciones de cada combinación de categoría

In [54]:
```
cross_tab_relative = pd.crosstab(
    data['FAMI_NUMLIBROS'],
    data['ESTU_DEDICACIONLECTURADIARIA'],
    normalize='index'   # Proporción por fila
)
cross_tab_relative['Total'] = cross_tab_relative.sum(axis=1)

cross_tab_relative
```

Out[54]:

ESTU_DEDICACIONLECTURADIARIA	30 minutos o menos	Entre 1 y 2 horas	Entre 30 y 60 minutos	Más de 2 horas	No leo por entretenimiento
FAMI_NUMLIBROS					
0 A 10 LIBROS	0.438385	0.074448	0.218019	0.028298	0.240850
11 A 25 LIBROS	0.381870	0.102858	0.302509	0.039091	0.173672
26 A 100 LIBROS	0.322139	0.141331	0.320195	0.062948	0.153388
MÁS DE 100 LIBROS	0.271548	0.171544	0.301988	0.107043	0.147876

También se pueden ver las tablas dinámicas como un "heatmap"

In [56]:
```
# Crear la tabla cruzada
cross_tab = pd.crosstab(data['FAMI_ESTRATOVIVIENDA'], data['ESTU_DEDICACIONLECTURAD

# Calcular las frecuencias relativas por fila (porcentaje de cada estrato)
cross_tab_pct = cross_tab.div(cross_tab.sum(axis=1), axis=0) * 100

# Obtener el orden de las categorías de lectura según su participación en el Estrat
orden_categorias = cross_tab_pct.loc['Estrato 1'].sort_values(ascending=False).inde

# Reordenar los niveles del índice de columnas
cross_tab_pct = cross_tab_pct.reindex(columns=orden_categorias)

# Crear el heatmap con las columnas reordenadas
sns.heatmap(cross_tab_pct, annot=True, cmap='Blues', fmt='.1f')
plt.title('Tabla dinámica de frecuencia relativa entre estrato y tiempo de dedicaci
```

```
plt.xlabel('Tiempo de dedicación de lectura diaria', fontsize=12)
plt.ylabel('Estrato de la vivienda', fontsize=12)
plt.show()
```

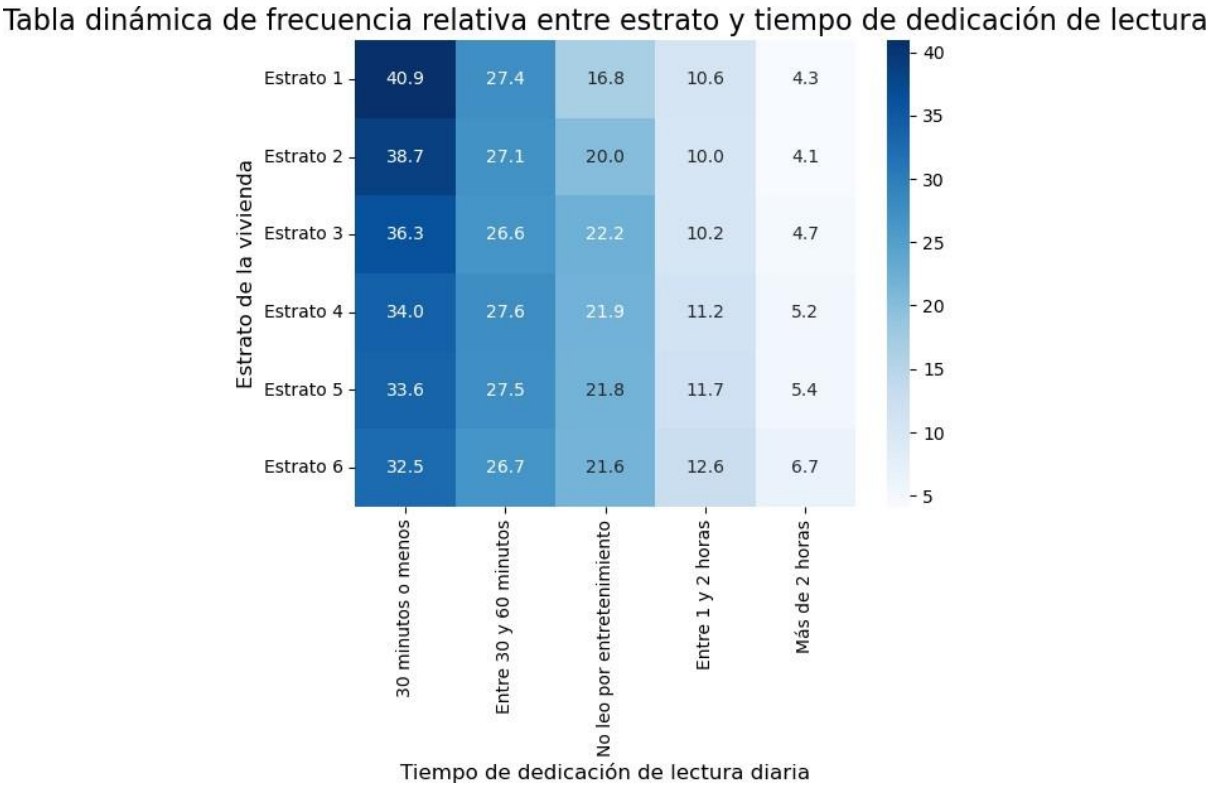


Gráfico de barras apilado

Un gráfico de barras que apila una variable dentro de otra para mostrar frecuencias y proporciones.

In [58]:
```
cross_tab = pd.crosstab(data['FAMI_NUMLIBROS'], data['FAMI_TIENECOMPUTADOR'])
cross_tab.plot(kind='bar', stacked=True, colormap='magma', figsize=(10, 6))
plt.title('Frecuencia y proporción de posesión de computador por rango número de li
plt.xlabel('Número de libros en casa', fontsize=12)
plt.ylabel('Frecuencia', fontsize=12)
plt.legend(title='Posee computador', fontsize=10)
plt.show()
```

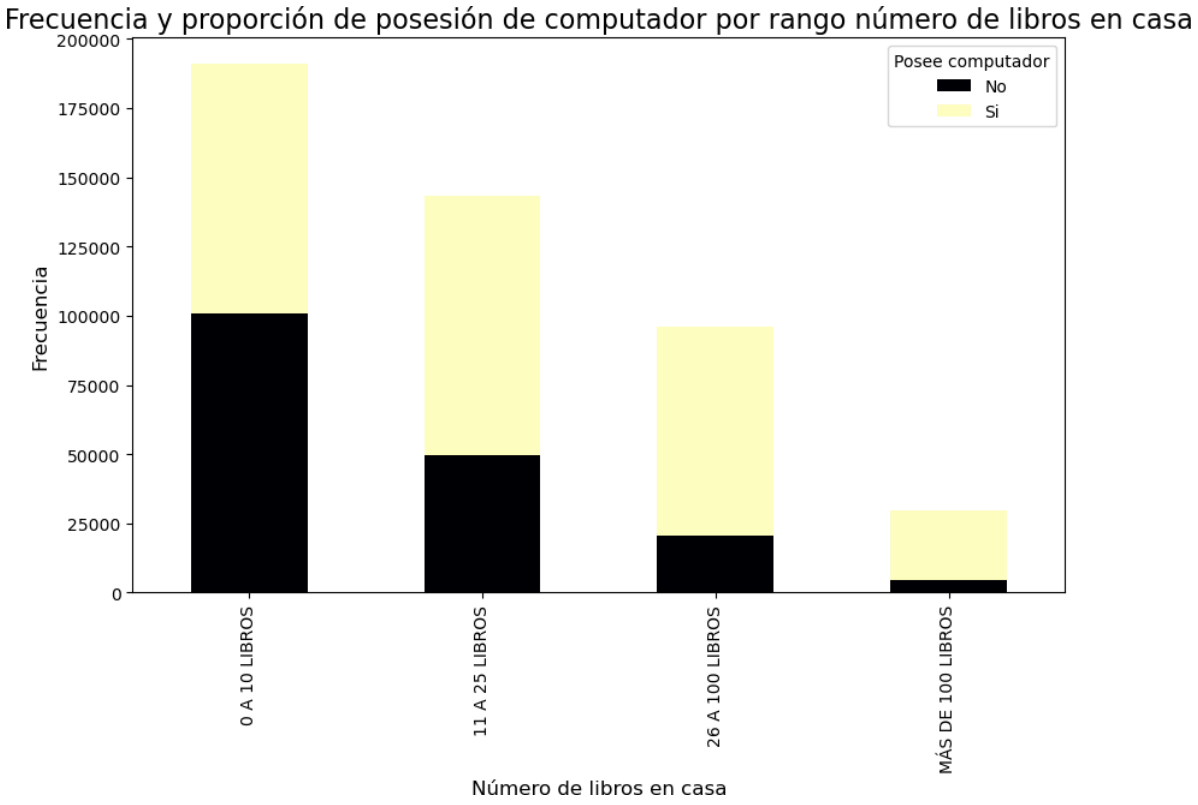
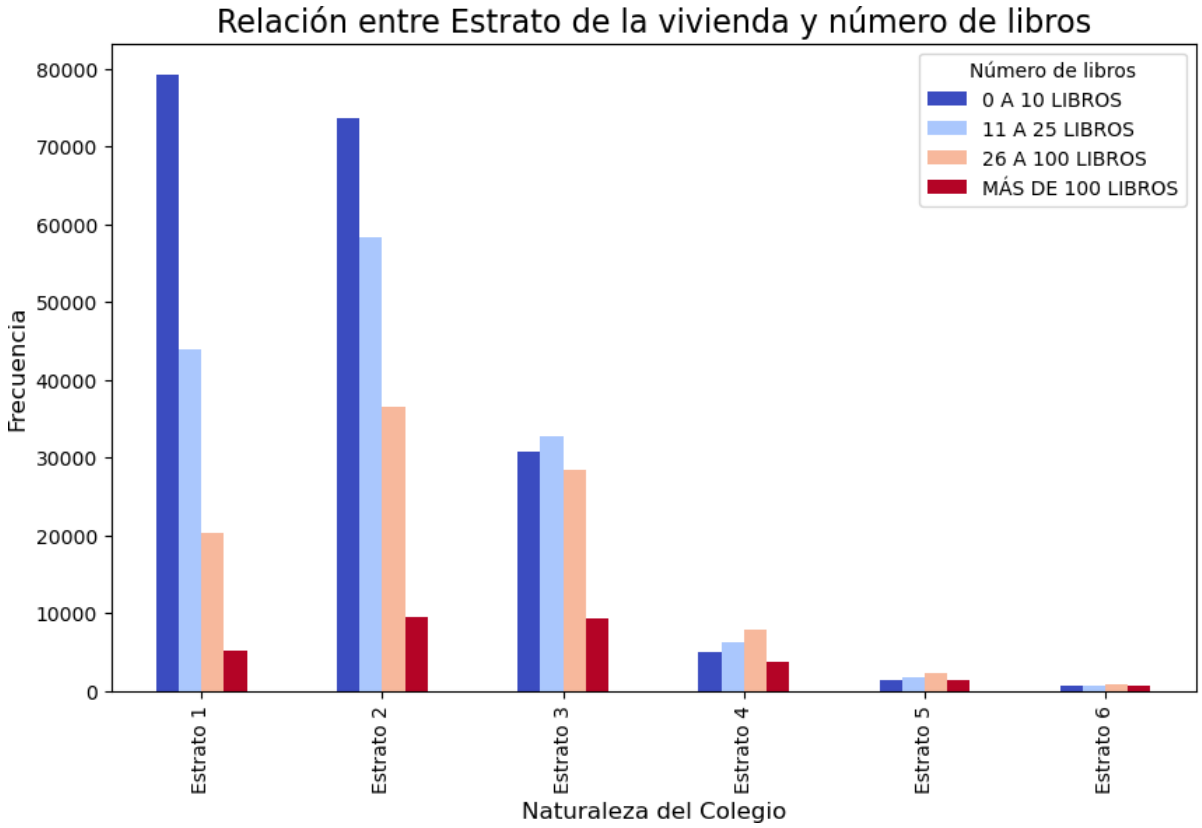


Gráfico de barras agrupado

Similar al gráfico de barras apilado, pero muestra las categorías lado a lado.

In [60]:
```
cross_tab = pd.crosstab(data['FAMI_ESTRATOVIVIENDA'], data['FAMI_NUMLIBROS'])
cross_tab.plot(kind='bar', colormap='coolwarm', figsize=(10, 6))
plt.title('Relación entre Estrato de la vivienda y número de libros', fontsize=16)
plt.xlabel('Naturaleza del Colegio', fontsize=12)
plt.ylabel('Frecuencia',  fontsize=12)
plt.legend(title='Número de libros', fontsize=10)
plt.show()
```



Visualización conjunta de variables cualitativas y cuantitativas

En esta subsección se hará una introducción sobre los principales métodos de visualización para variables cuantitativas y cualitativas en conjunto. Entre las principales herramientas se encuentran:

1. Bar-Plot con promedio y desviación estándar por categoría

2. Box-Plot por categoría

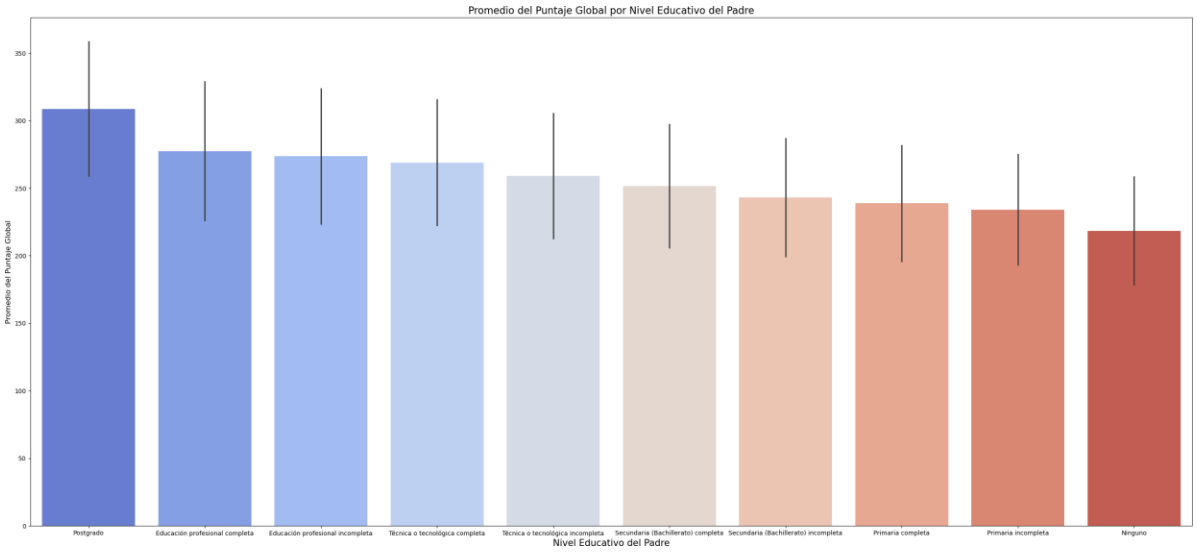
3. Histogramas agrupados por categoría

In [62]:
```
import warnings
warnings.filterwarnings("ignore")
# Filtrar el DataFrame para eliminar categorías específicas en 'FAMI_EDUCACIONPADRE
data_1 = data[~data['FAMI_EDUCACIONPADRE'].isin(['No Aplica', 'No sabe'])]
```

Bar-Plot con promedio y desviación estándar por categoría

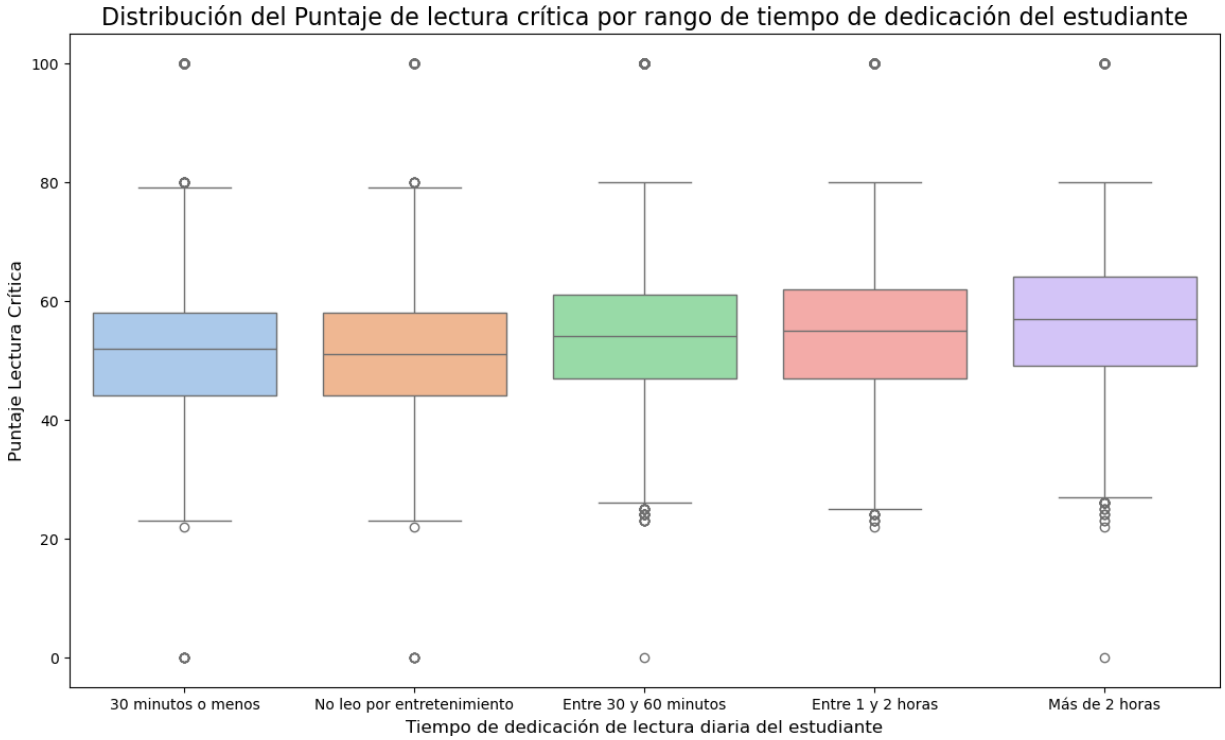
In [64]:
```
# Calcular los promedios para ordenar las categorías
order = data_1.groupby('FAMI_EDUCACIONPADRE')['PUNT_GLOBAL'].mean().sort_values(asc

# Crear el gráfico de barras ordenado
```

```
plt.figure(figsize=(34, 15))
sns.barplot(
    x=data_1['FAMI_EDUCACIONPADRE'],
    y=data_1['PUNT_GLOBAL'],
    errorbar='sd',
    palette='coolwarm',
    order=order  # Ordenar las barras por el promedio
)

# Personalización del gráfico
plt.title('Promedio del Puntaje Global por Nivel Educativo del Padre', fontsize=16)
plt.xlabel('Nivel Educativo del Padre', fontsize=15)
plt.ylabel('Promedio del Puntaje Global', fontsize=12)
plt.show()
```



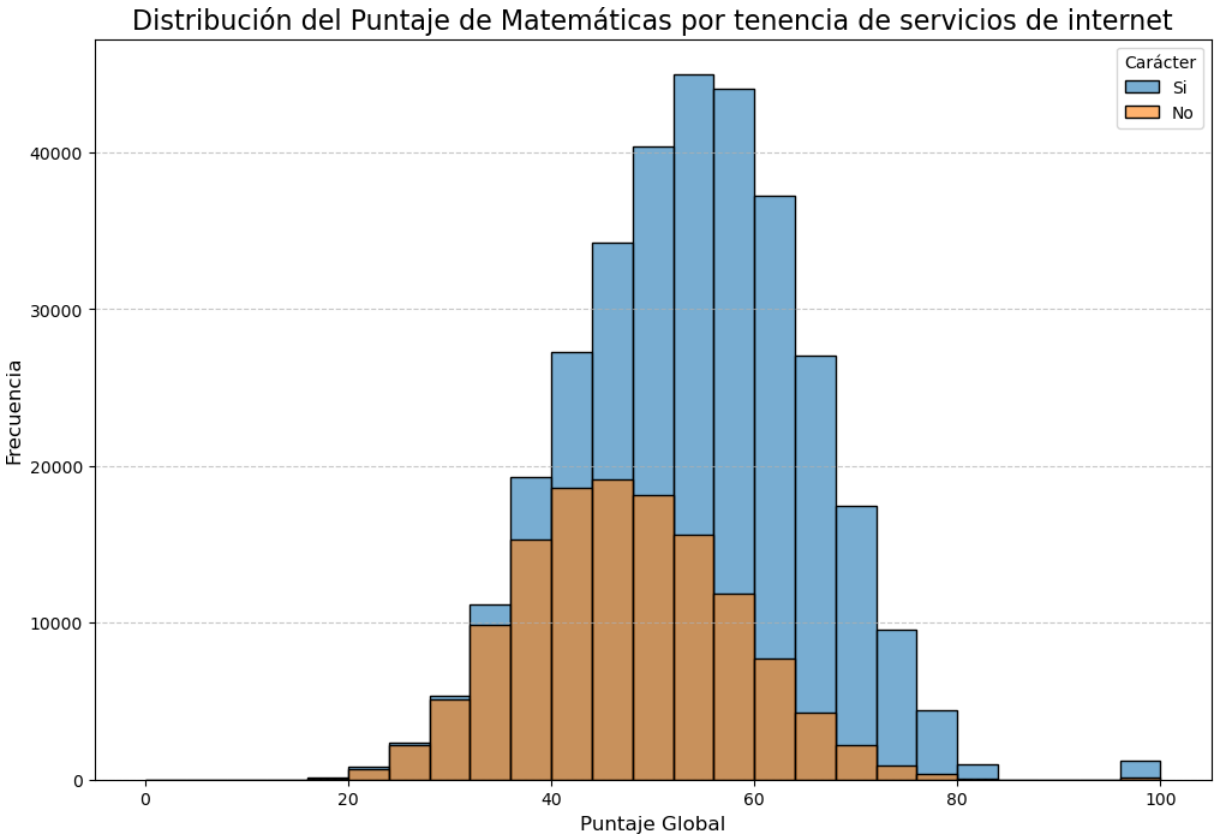
Box-Plot por categoría

In [66]:
```
plt.figure(figsize=(14, 8))
sns.boxplot(x=data['ESTU_DEDICACIONLECTURADIARIA'],  y=data['PUNT_LECTURA_CRITICA'],
plt.title('Distribución del Puntaje de lectura crítica por rango de tiempo de dedic
plt.xlabel('Tiempo de dedicación de lectura diaria del estudiante', fontsize=12)
plt.ylabel('Puntaje Lectura Crítica', fontsize=12)
plt.show()
```



Histogramas agrupados por categoría

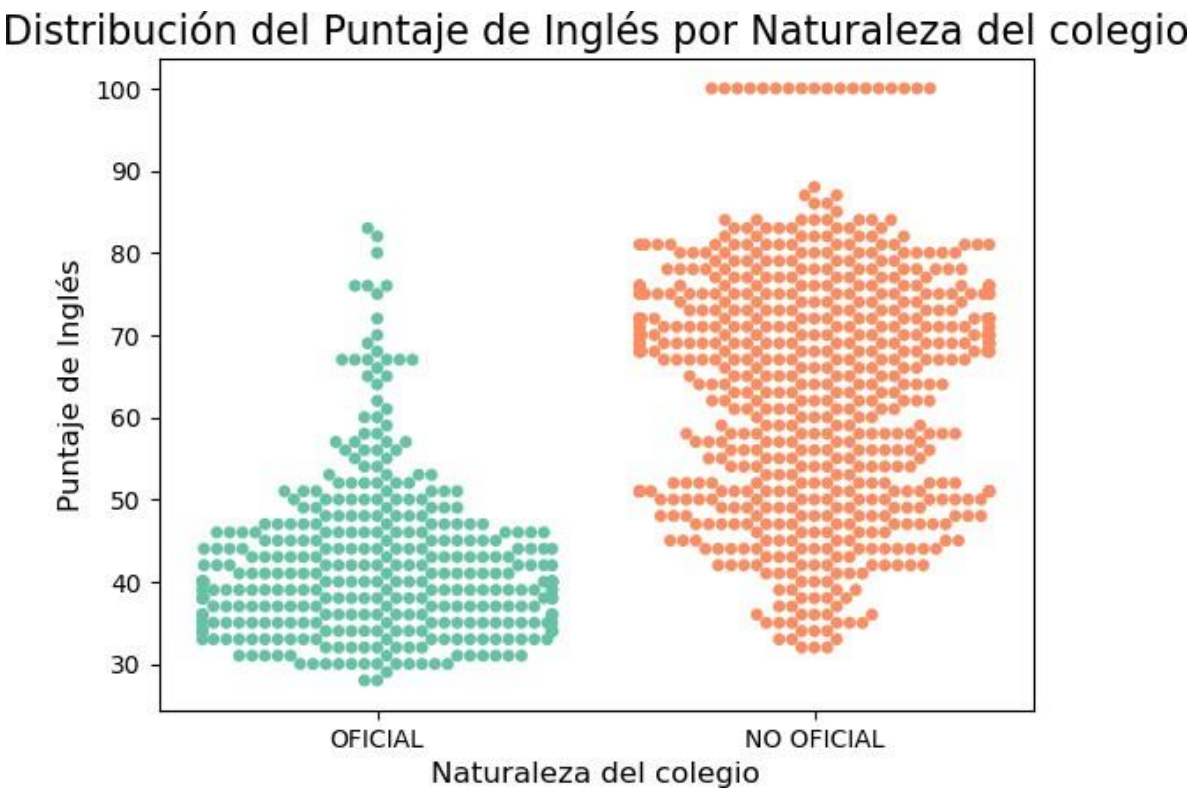
In [68]:
```
# Crear histogramas agrupados sin KDE y ajustar el eje Y para el puntaje global
plt.figure(figsize=(12, 8))  # Ajustar tamaño del gráfico

for categoria in data['FAMI_TIENEINTERNET'].unique():
    subset = data[data['FAMI_TIENEINTERNET'] == categoria]
    sns.histplot(
        subset['PUNT_MATEMATICAS'],
        label=categoria,
        alpha=0.6,
        bins=25  # Ajustar número de barras
    )

# Personalización del gráfico
plt.title('Distribución del Puntaje de Matemáticas por tenencia de servicios de int
plt.xlabel('Puntaje Global', fontsize=12)
plt.ylabel('Frecuencia', fontsize=12)
plt.legend(title='Carácter', fontsize=10)
plt.grid(axis='y', linestyle='--', alpha=0.7)  # Añadir líneas de referencia en el
plt.show()
```



Swarm-Plot

Un Swarm-Plot es un tipo de gráfico de visualización de datos que te permite observar la distribución de un conjunto de datos numéricos, categorizados por una variable categórica.

In [70]:
```
sns.swarmplot(x=f_data_sample['COLE_NATURALEZA'],  y=f_data_sample['PUNT_INGLES'],  p
plt.title('Distribución del Puntaje de Inglés por Naturaleza del colegio', fontsize
plt.xlabel('Naturaleza del colegio', fontsize=12)
plt.ylabel('Puntaje de Inglés', fontsize=12)
plt.show()
```



Conclusión

El Análisis Exploratorio de Datos (AED) constituye un paso esencial en todo proceso riguroso de análisis cuantitativo. A través de esta guía, se ha mostrado cómo Python —con bibliotecas como pandas, matplotlib y seaborn— permite implementar de forma accesible y reproducible técnicas de inspección, limpieza, visualización y síntesis estadística de datos. Trabajando con información de las Pruebas Saber 11 (2020), se exploraron patrones académicos, familiares y socioeconómicos de los estudiantes en Colombia, demostrando cómo el AED ayuda a conocer el comportamiento general de un conjunto de datos. Más allá de los procedimientos técnicos, esta guía busca contribuir al desarrollo de un pensamiento analítico, basado en la evidencia y con sensibilidad por el contexto social de los datos. En el caso de la economía, este tipo de exploración preliminar permite enriquecer el análisis empírico, identificar patrones y comprender mejor las relaciones complejas entre variables económicas y sociales.

En resumen, el AED no es solo una etapa metodológica, sino una herramienta de observación crítica que debe acompañar cualquier proceso de investigación empírica. Dominar sus fundamentos es clave para construir análisis responsables, transparentes y útiles para la toma de decisiones.