Análisis y visualización de datos | Lorenzo Tomás Diez

Actividad 4

Situación

- Continuás armando el informe para esta empresa multinacional. Ahora que estás familiarizado con las distintas bases de datos y sus variables, podés empezar a plantear algunas hipótesis usando lo que aprendimos en este módulo.
- Como en cualquier proyecto de datos, es importante que desde el comienzo vayas pensando (aunque sea en términos generales) cómo vas a presentar los resultados a tu audiencia, tus clientes. En este caso, es una audiencia que tiene mucho conocimiento de su industria en particular, pero no tiene tanto conocimiento técnico de análisis de datos. Ni, como mencionamos, de la economía y la sociedad de nuestro país.

Requerimientos

- Para realizar esta actividad descargá el archivo ZIP, que encontrarás en la plataforma al final de este documento.

Consignas

1. En función de lo que ya conocés de estas bases, planteá dos hipótesis, relacionándolas con las variables que están en las bases, recordá que tienen que estar en forma de afirmaciones.
2. Hacé un análisis exploratorio en base a esas hipótesis y explicá si en principio la evidencia de los datos respalda esas hipótesis o no.
3. Escribí un ejemplo de una afirmación que no permitan hacer los datos disponibles en estas bases.
4. Describí dos o tres aspectos que tendrías en cuenta para comunicarlo a una audiencia no técnica.

Parte 1: Hipotesis y analisis exploratorio de los datos

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

# Cargar los datos
df_empleos = pd.read_csv('./empleo.csv', encoding='latin-1')
df_exportaciones = pd.read_csv('./exportaciones.csv', encoding='latin-1')
```

```
df_empleos.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3696 entries, 0 to 3695
Data columns (total 4 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   provincia                   3696 non-null   object
 1   anio                        3696 non-null   int64
 2   mes                         3696 non-null   object
 3   empleados_registrados_miles  3696 non-null   float64
dtypes: float64(1), int64(1), object(2)
memory usage: 115.6+ KB
```

```
df_exportaciones.head()
```

	rubro	provincia	anio	value
0	Total	Total País	2005	40386.762
1	Total	Buenos Aires	2005	15626.865
2	Total	Capital Federal	2005	271.295
3	Total	Córdoba	2005	4452.178
4	Total	Entre Ríos	2005	815.341

Next steps: Generate code with df_exportaciones | View recommended plots | New interactive sheet

```
df_exportaciones['provincia'].unique()
```

```
array(['Total País', 'Buenos Aires', 'Capital Federal', 'Córdoba',
       'Entre Ríos', 'La Pampa', 'Santa Fe', 'Chubut', 'Neuquén',
       'Río Negro', 'Santa Cruz', 'Tierra del Fuego', 'Catamarca',
```

```
       'Jujuy', 'La Rioja', 'Salta', 'Santiago del Estero', 'Tucumán',
       'Mendoza', 'San Juan', 'San Luis', 'Chaco', 'Corrientes',
       'Formosa', 'Misiones'], dtype=object)
```

```
df_exportaciones = df_exportaciones[df_exportaciones['provincia'] != 'Total País']
df_exportaciones.head(10)
```

	rubro	provincia	anio	value
1	Total	Buenos Aires	2005	15626.865
2	Total	Capital Federal	2005	271.295
3	Total	Córdoba	2005	4452.178
4	Total	Entre Ríos	2005	815.341
5	Total	La Pampa	2005	242.874
6	Total	Santa Fe	2005	7720.431
7	Total	Chubut	2005	2038.589
8	Total	Neuquén	2005	900.583
9	Total	Río Negro	2005	415.902
10	Total	Santa Cruz	2005	858.361

Next steps: Generate code with df_exportaciones | View recommended plots | New interactive sheet

```
df_empleos['provincia'].unique()
```

```
array(['Buenos Aires', 'Capital Federal', 'Catamarca', 'Chaco', 'Chubut',
       'Córdoba', 'Corrientes', 'Entre Ríos', 'Formosa', 'Jujuy',
       'La Pampa', 'La Rioja', 'Mendoza', 'Misiones', 'Neuquén',
       'Río Negro', 'Salta', 'San Juan', 'San Luis', 'Santa Cruz',
       'Santa Fe', 'Santiago del Estero', 'Tierra del Fuego', 'Tucumán'],
      dtype=object)
```

Hipótesis 1: Existe una correlación positiva significativa entre el valor total de las exportaciones y el número de empleados registrados en las provincias argentinas para cada año.

El analisis exploratorio que vamos a realizar es el siguiente:

1. Preparar los datos:
   - Filtrar los datos de exportaciones y empleo para el mismo año (el más reciente disponible).
   - Agrupar los datos de exportaciones por provincia y calcular el total de exportaciones.
   - Calcular el promedio anual de empleados registrados por provincia.
2. Visualizar la relación:
   - Crear un gráfico de dispersión con exportaciones en el eje X y empleo en el eje Y.
   - Añadir una línea de tendencia para visualizar la correlación.
3. Calcular la correlación:
   - Utilizar el coeficiente de correlación de Pearson para cuantificar la relación.
4. Analizar outliers:
   - Identificar provincias que se desvían significativamente de la tendencia general.
5. Realizar un análisis temporal:
   - Si hay datos disponibles para varios años, analizar cómo ha evolucionado esta relación en el tiempo.

```
df_empleos['anio'].unique()
```

```
array([2009, 2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017, 2018, 2019,
       2020, 2021])
```

```
df_exportaciones['anio'].unique()
```

```
array([2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014, 2015,
       2016, 2017, 2018, 2019, 2020])
```

1 - Preparar los datos:

- Vamos a tomar el año 2020 para realizar el analisis, ya que es el año el cual tiene datos disponibles para ambas bases.
- En empleos, vamos a tomar la media del ano 2020 para que sea un dato representativo. Finalmente eliminamos la columna 'mes' ya que no vamos a utilizarla.
- En exportaciones, vamos a tomar el rubro 'Total' para luego eliminar la columna 'rubro'. Ademas vamos a renombrar la columna 'value' a 'exportaciones' para que sea mas claro.

```
df_empleos_2020 = df_empleos[df_empleos['anio'] == 2020]
df_empleos_2020 = df_empleos_2020.groupby('provincia').agg({'empleados_registrados_miles': 'mean'}).reset_index()
df_empleos_2020.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 2 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   provincia                   24 non-null     object
 1   empleados_registrados_miles  24 non-null     float64
dtypes: float64(1), object(1)
memory usage: 512.0+ bytes
```

```
df_exportaciones_2020 = df_exportaciones[df_exportaciones['anio'] == 2020]
df_exportaciones_2020 = df_exportaciones_2020[df_exportaciones_2020['rubro'] == 'Total']
df_exportaciones_2020.drop(columns=['anio', 'rubro'], inplace=True)
df_exportaciones_2020.rename(columns={'value': 'exportaciones'}, inplace=True)
df_exportaciones_2020['exportaciones'] = df_exportaciones_2020['exportaciones'].str.replace(',', '').astype(float)
df_exportaciones_2020.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 24 entries, 1876 to 1899
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   provincia      24 non-null     object
 1   exportaciones  24 non-null     float64
dtypes: float64(1), object(1)
memory usage: 576.0+ bytes
```

Finalmente:

- Vamos a hacer un merge de las bases, con provincia y anio como variables en comun.

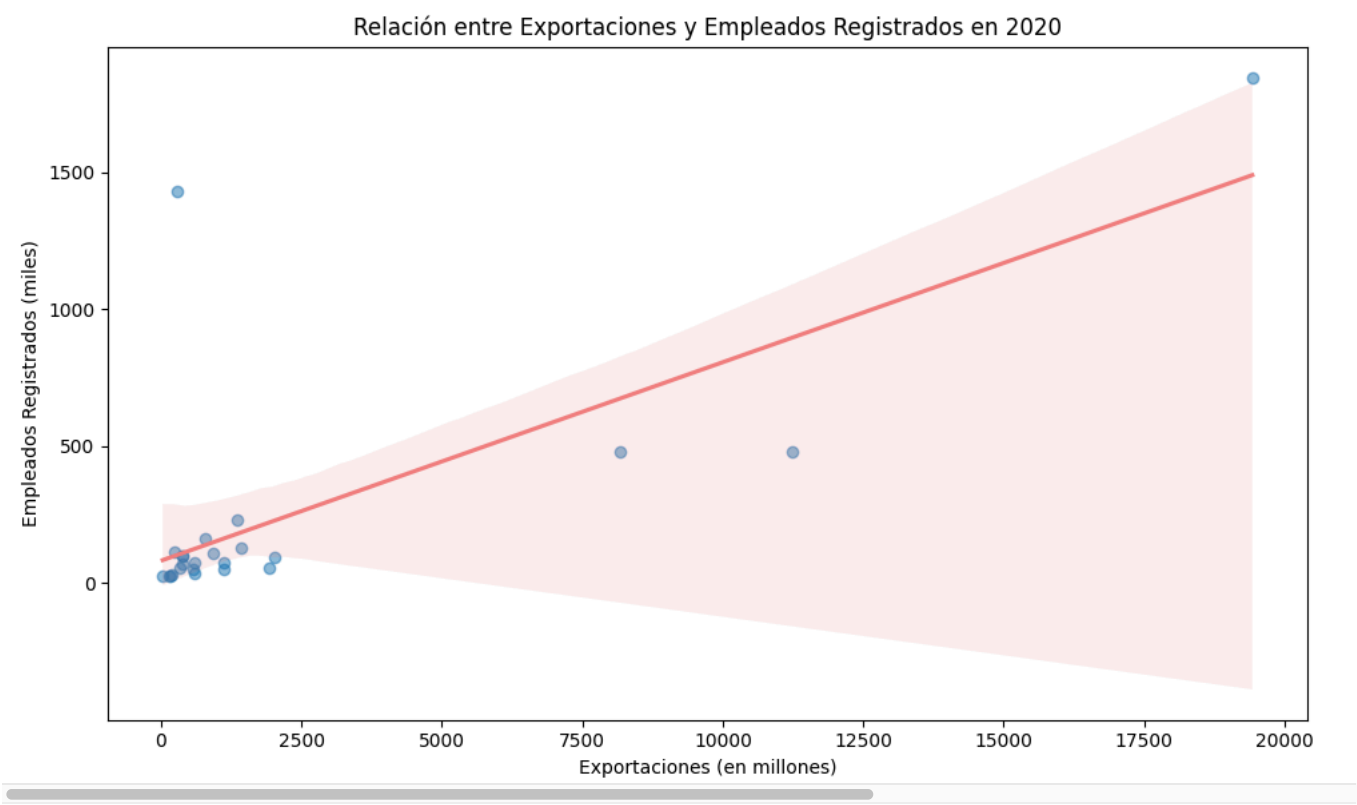
```
df_expo_empleo_2020 = pd.merge(df_exportaciones_2020, df_empleos_2020, on=['provincia'])
df_expo_empleo_2020.head()
```

	provincia	exportaciones	empleados_registrados_miles
0	Buenos Aires	19428.334	1844.903833
1	Capital Federal	280.854	1431.688833
2	Córdoba	8162.930	477.492000
3	Entre Ríos	1425.285	129.005500
4	La Pampa	592.325	35.653083

Next steps: Generate code with df_expo_empleo_2020 | View recommended plots | New interactive sheet

2 - Visualizar la relación:

```
plt.figure(figsize=(10, 6))

ax = sns.regplot(x='exportaciones', y='empleados_registrados_miles',
    data=df_expo_empleo_2020,
    scatter_kws={'alpha':0.5},
    line_kws={'color': 'lightcoral'})
ax.set_title('Relación entre Exportaciones y Empleados Registrados en 2020')
ax.set_xlabel('Exportaciones (en millones)')
ax.set_ylabel('Empleados Registrados (miles)')

plt.tight_layout()
plt.show()
```



3 - Calcular la correlación:

```
x = df_expo_empleo_2020['exportaciones']
y = df_expo_empleo_2020['empleados_registrados_miles']

matriz_correlacion = np.corrcoef(x, y)
matriz_correlacion
```

```
array([[1.       , 0.7246479],
       [0.7246479, 1.       ]])
```

El valor 0.7246 sugiere una correlación positiva moderada a fuerte entre las dos variables. Pero puede estar influenciado por otros factores como la población de la provincia, la cantidad de empresas, etc.

Vamos a analizar los valores atipicos:

4 - Análisis de valores atipicos:

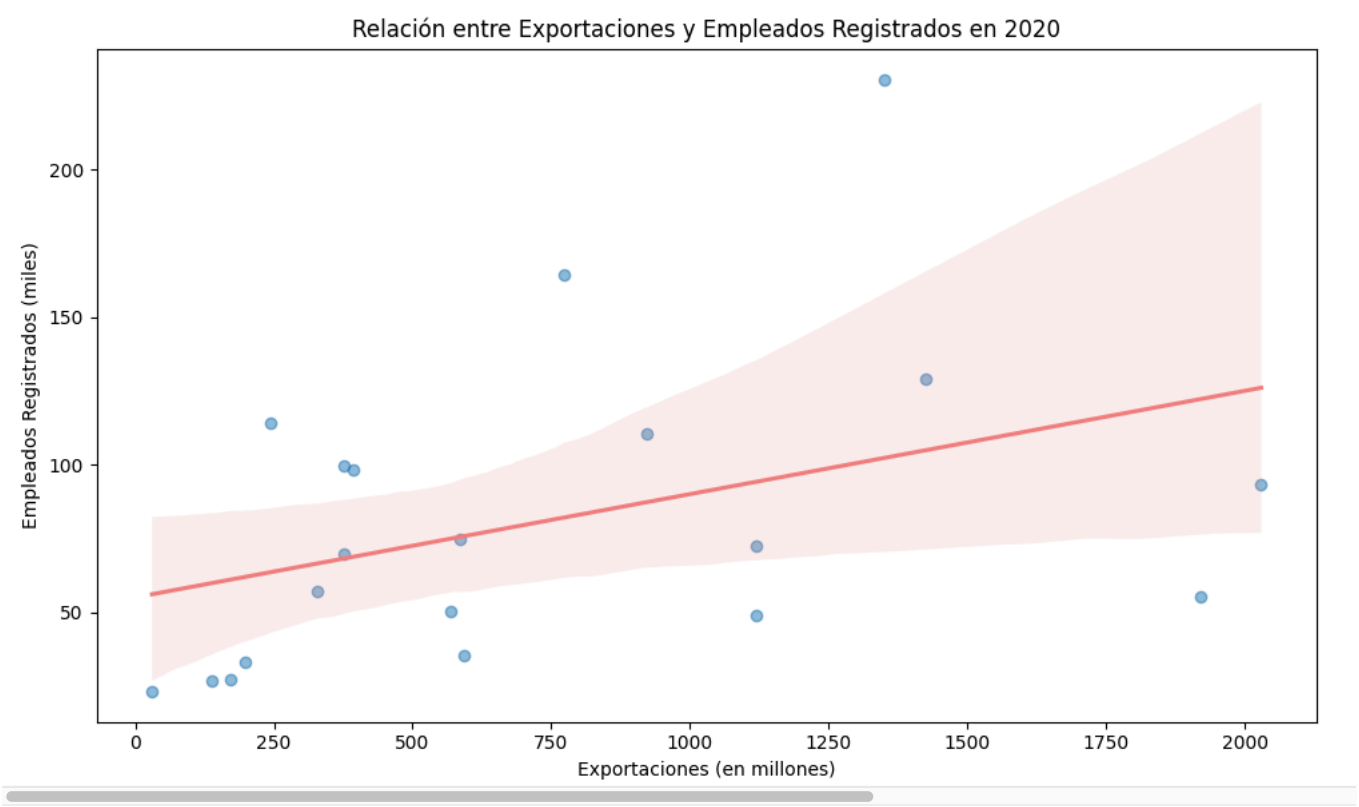
- Vamos a eliminar los valores atipicos, eliminando las provincias que se desvían significativamente de la tendencia general.

```
Q1 = df_expo_empleo_2020['empleados_registrados_miles'].quantile(0.25)
Q3 = df_expo_empleo_2020['empleados_registrados_miles'].quantile(0.75)
IQR = Q3 - Q1
limite_inferior = Q1 - 1.5 * IQR
limite_superior = Q3 + 1.5 * IQR
df_expo_empleo_2020_outliers = df_expo_empleo_2020[
  (df_expo_empleo_2020['empleados_registrados_miles'] >= limite_inferior) &
  (df_expo_empleo_2020['empleados_registrados_miles'] <= limite_superior)
]
df_expo_empleo_2020_outliers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 20 entries, 3 to 23
Data columns (total 3 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   provincia                   20 non-null     object
 1   exportaciones               20 non-null     float64
 2   empleados_registrados_miles  20 non-null     float64
dtypes: float64(2), object(1)
memory usage: 640.0+ bytes
```

Vemos que se han eliminado 4 provincias en nuestro analisis. Vamos a volver a graficar la relacion entre exportaciones e empleados registrados, pero esta vez sin los outliers.

```
plt.figure(figsize=(10, 6))

ax = sns.regplot(x='exportaciones', y='empleados_registrados_miles',
    data=df_expo_empleo_2020_outliers,
    scatter_kws={'alpha':0.5},
    line_kws={'color': 'lightcoral'})
ax.set_title('Relación entre Exportaciones y Empleados Registrados en 2020')
ax.set_xlabel('Exportaciones (en millones)')
ax.set_ylabel('Empleados Registrados (miles)')

plt.tight_layout()
plt.show()
```



Volvemos a calcular la correlacion:

```
x = df_expo_empleo_2020_outliers['exportaciones']
y = df_expo_empleo_2020_outliers['empleados_registrados_miles']

matriz_correlacion = np.corrcoef(x, y)
matriz_correlacion
```

```
array([[1.        , 0.39612755],
       [0.39612755, 1.        ]])
```

El resultado es de 0.3951, lo cual es una correlacion positiva baja. Esto nos indica que los valores atipicos estaban influyendo en la correlacion.

5 - Análisis temporal:

- Ahora vamos a aplicar la misma logica pero con mas anos, para verificar que nuestro anilisis no este sesgado por un solo ano.
- Vamos a tomar desde 2009 a 2020 ya que son los anios que comparten ambas bases.

```
df_expo_total = df_exportaciones[df_exportaciones['rubro'] == 'Total'].copy()
df_expo_total.drop(columns=['rubro'], inplace=True)
df_expo_total['value'] = df_expo_total['value'].str.replace(',', '').astype(float)
df_expo_total.rename(columns={'value': 'exportaciones'}, inplace=True)
df_expo_total = df_expo_total[df_expo_total['anio'].isin(range(2009, 2021))]
df_expo_total.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 288 entries, 501 to 1899
```

```
Data columns (total 3 columns):
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   provincia      288 non-null    object
 1   anio           288 non-null    int64
 2   exportaciones  288 non-null    float64
dtypes: float64(1), int64(1), object(1)
memory usage: 9.0+ KB
```

```
df_empleo_total = df_empleos[df_empleos['anio'].isin(range(2009, 2021))].copy()
df_empleo_total = df_empleo_total.groupby(['anio', 'provincia']).agg({'empleados_registrados_miles': 'mean'}).reset_index()
df_empleo_total.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 288 entries, 0 to 287
Data columns (total 3 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   anio                         288 non-null    int64
 1   provincia                    288 non-null    object
 2   empleados_registrados_miles  288 non-null    float64
dtypes: float64(1), int64(1), object(1)
memory usage: 6.9+ KB
```

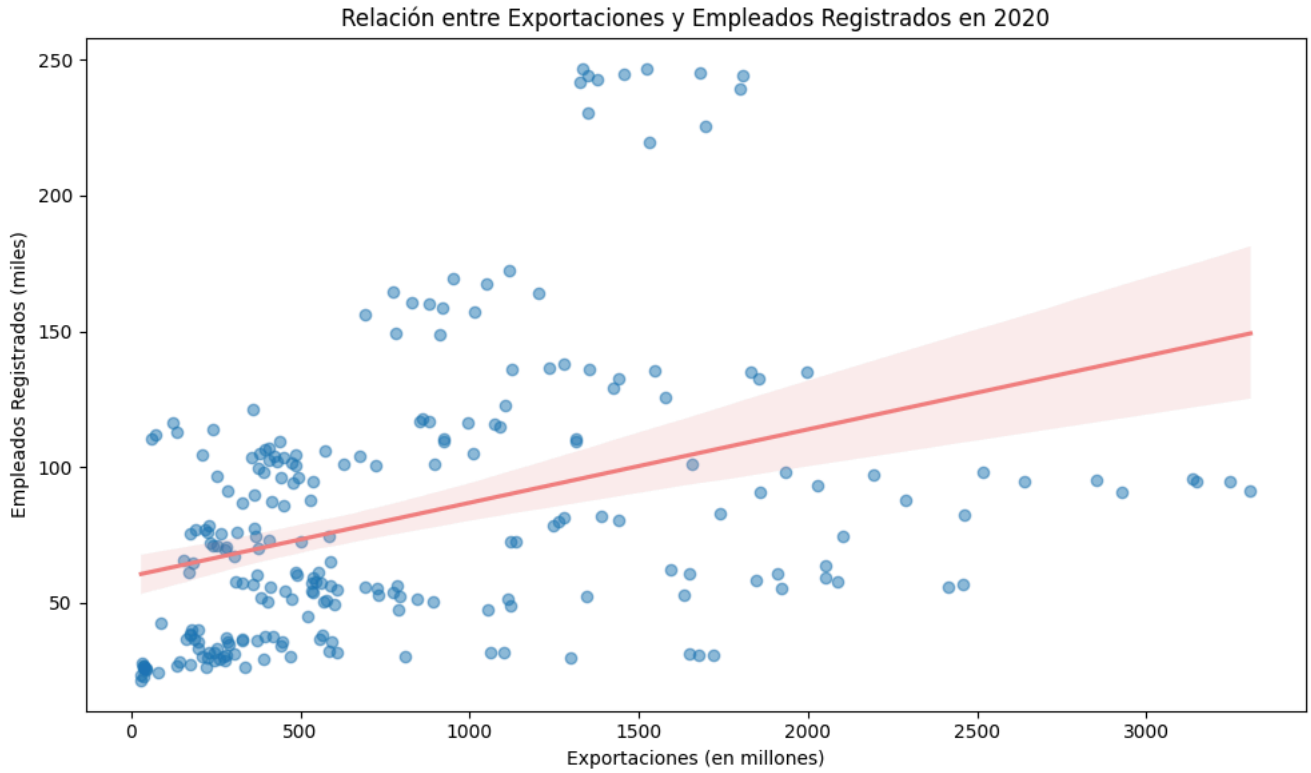
```
df_expo_empleo_total = pd.merge(df_expo_total, df_empleo_total, on=['anio', 'provincia'])
df_expo_empleo_total.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 288 entries, 0 to 287
Data columns (total 4 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   provincia                    288 non-null    object
 1   anio                         288 non-null    int64
 2   exportaciones                288 non-null    float64
 3   empleados_registrados_miles  288 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 9.1+ KB
```

```
Q1_empleo_expo_total = df_expo_empleo_total['empleados_registrados_miles'].quantile(0.25)
Q3_empleo_expo_total = df_expo_empleo_total['empleados_registrados_miles'].quantile(0.75)

IQR_empleo_expo_total = Q3_empleo_expo_total - Q1_empleo_expo_total

limite_inferior_empleo_expo_total = Q1_empleo_expo_total - 1.5 * IQR_empleo_expo_total
limite_superior_empleo_expo_total = Q3_empleo_expo_total + 1.5 * IQR_empleo_expo_total

df_expo_empleo_total_outliers = df_expo_empleo_total[
  (df_expo_empleo_total['empleados_registrados_miles'] >= limite_inferior_empleo_expo_total) &
  (df_expo_empleo_total['empleados_registrados_miles'] <= limite_superior_empleo_expo_total)
]
df_expo_empleo_total_outliers.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 240 entries, 3 to 287
Data columns (total 4 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   provincia                    240 non-null    object
 1   anio                         240 non-null    int64
 2   exportaciones                240 non-null    float64
 3   empleados_registrados_miles  240 non-null    float64
dtypes: float64(2), int64(1), object(1)
memory usage: 9.4+ KB
```

```
plt.figure(figsize=(10, 6))

ax = sns.regplot(x='exportaciones', y='empleados_registrados_miles',
    data=df_expo_empleo_total_outliers,
    scatter_kws={'alpha':0.5},
    line_kws={'color': 'lightcoral'})
ax.set_title('Relación entre Exportaciones y Empleados Registrados en 2020')
ax.set_xlabel('Exportaciones (en millones)')
ax.set_ylabel('Empleados Registrados (miles)')

plt.tight_layout()
plt.show()
```



Verificamos la correlacion:

```
x = df_expo_empleo_total_outliers['exportaciones']
y = df_expo_empleo_total_outliers['empleados_registrados_miles']

matriz_correlacion_total = np.corrcoef(x, y)
matriz_correlacion_total
```

```
array([[1.        , 0.37440203],
       [0.37440203, 1.        ]])
```

El valor es 0.3744, lo cual es una correlacion positiva baja.

Conclusion:

- La correlacion entre exportaciones e empleados registrados es baja, lo cual nos indica que no hay una relacion lineal estrecha entre ambas variables.
- El analisis temporal reafirma que la correlacion es baja y no presenta una tendencia clara a lo largo de los anios.
- Por lo tanto, no podemos afirmar que exista una correlacion significativa entre el valor total de las exportaciones y el número de empleados registrados en las provincias argentinas para cada año.

Hipótesis 2: Existe una variación estacional significativa en el número de empleos registrados en las provincias argentinas, con un aumento durante los meses de primavera (septiembre a noviembre) y una disminución durante los meses de invierno (junio a agosto), cada anio.

1. Preparar los datos:
   - Filtrar los datos por provincia y mes: Asegúrate de tener los datos mensuales de empleos registrados en las provincias.
   - Agrupar los datos por provincia y mes: Esto te permitirá analizar las tendencias a lo largo del tiempo para cada provincia.
   - Definir las estaciones: Crea una nueva columna que clasifique los meses en sus respectivas estaciones (invierno, primavera, verano, otoño).
2. Visualizar la variación estacional:
   - Crear un gráfico de líneas o barras para visualizar el número de empleados registrados por provincia y estación del año.
   - Comparar las estaciones de primavera e invierno para ver las diferencias en la cantidad de empleados.
3. Calcular la variación estacional:

- Calcular el promedio de empleados registrados para cada estación del año (primavera, invierno, verano, otoño) por provincia.
- Comparar los promedios de primavera e invierno para verificar si existe una diferencia significativa.

4. Analizar outliers:

- Identificar provincias que se desvíen significativamente de la tendencia general en cuanto a la variación estacional.

1 - Preparar los datos:

- Vamos a tomar los datos de 2019-2021 para que sea un periodo representativo.
- Vamos a crear la columna 'estacion' que clasifique los meses en sus respectivas estaciones.
- Vamos a eliminar la columna 'mes' ya que no vamos a utilizarla.
- Vamos a tomar solo las estaciones de primavera e invierno para que sea mas facil de visualizar.
- Finalmente vamos a calcular el promedio de empleados registrados para cada estación del año (primavera, invierno, verano, otoño) por provincia.

```
estaciones = {
  'verano': ['dic', 'ene', 'feb'],
  'otonio': ['mar', 'abr', 'may'],
  'invierno': ['jun', 'jul', 'ago'],
  'primavera': ['sep', 'oct', 'nov']
}
def clasificar_estacion(mes_ingresado):
  for estacion, meses in estaciones.items():
    for mes in meses:
      if mes in mes_ingresado:
        return estacion
  return 'desconocido'

test_estacion = clasificar_estacion('mar-21')
test_estacion
```

```
'otonio'
```

```
df_empleos_estacion_2019_2021 = df_empleos[df_empleos['anio'].isin([2019, 2020, 2021])].copy()
df_empleos_estacion_2019_2021['estacion'] = df_empleos_estacion_2019_2021['mes'].apply(clasificar_estacion)
df_empleos_estacion_2019_2021 = df_empleos_estacion_2019_2021[df_empleos_estacion_2019_2021['estacion'].isin(['primavera', '
df_empleos_estacion_2019_2021.drop(columns=['mes'], inplace=True)
df_empleos_estacion_2019_2021 = df_empleos_estacion_2019_2021.groupby(['provincia', 'estacion']).agg({'empleados_registrados
df_empleos_estacion_2019_2021.head(10)
```

	provincia	estacion	empleados_registrados_miles
0	Buenos Aires	invierno	1850.6430
1	Buenos Aires	primavera	1868.7000
2	Capital Federal	invierno	1415.7830
3	Capital Federal	primavera	1428.7080
4	Catamarca	invierno	27.6990
5	Catamarca	primavera	27.3905
6	Chaco	invierno	72.1240
7	Chaco	primavera	72.2460
8	Chubut	invierno	93.6270
9	Chubut	primavera	93.7760

Next steps: Generate code with df_empleos_estacion_2019_2021 | View recommended plots | New interactive sheet

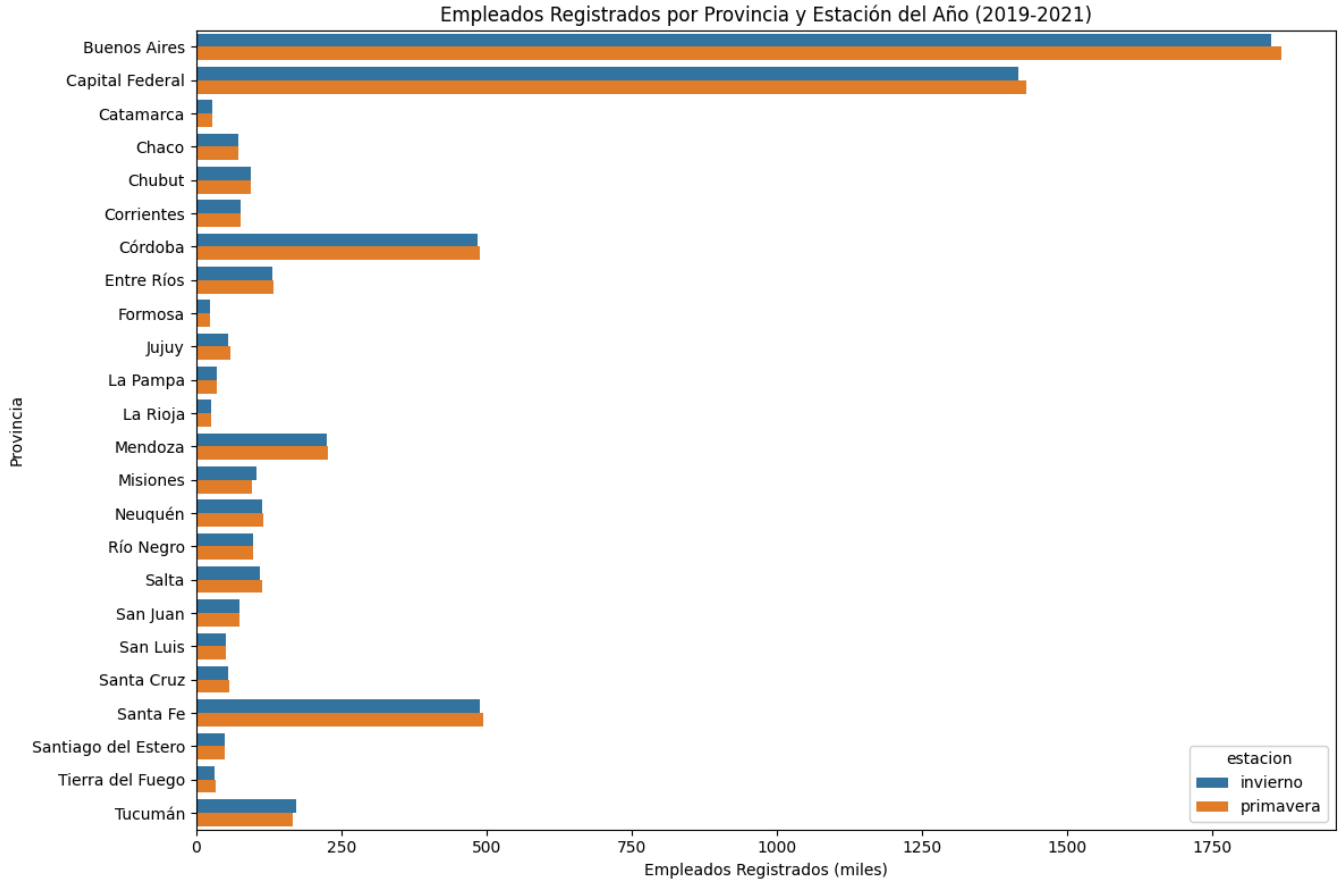
2 - Visualizar la variación estacional:

- Vamos a crear un grafico de barras para visualizar el numero de empleados registrados por provincia y estacion.

```
plt.figure(figsize=(12, 8))

ax = sns.barplot(x='empleados_registrados_miles', y='provincia', hue='estacion', data=df_empleos_estacion_2019_2021)
ax.set_title('Empleados Registrados por Provincia y Estación del Año (2019-2021)')
ax.set_xlabel('Empleados Registrados (miles)')
ax.set_ylabel('Provincia')
```

```
plt.tight_layout()
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, yo
  data_subset = grouped_data.get_group(pd_key)
/usr/local/lib/python3.10/dist-packages/seaborn/_base.py:949: FutureWarning: When grouping with a length-1 list-like, yo
  data_subset = grouped_data.get_group(pd_key)
```



3 - Calcular la variación estacional

- Vamos a comparar los promedios de primavera e invierno para verificar si existe una diferencia significativa.

Double-click (or enter) to edit

```
# Calcular el promedio de empleados registrados para primavera e invierno
promedios_estacionales = df_empleos_estacion_2019_2021.groupby('estacion').agg({'empleados_registrados_miles': 'mean'}).rese
promedios_estacionales
```

	estacion	empleados_registrados_miles
0	invierno	244.646417
1	primavera	246.495396

Next steps: Generate code with promedios_estacionales | View recommended plots | New interactive sheet

Conclusion:

- Existe una diferencia a favor de la estacion de primavera.
- La diferencia que existe, no es muy significativa, lo cual nos indica que la variacion estacional no es muy pronunciada.

Parte 2: Afirmaciones que no permiten hacer los datos disponibles

Afirmacion 1: Las provincias con mayor cantidad de habitantes tienden a tener una mayor cantidad de empleos registrados.

Afirmacion 2: Las provincias con mayor cantidad de habitantes tienden a tener mayor cantidad de exportaciones.

Parte 3: Comunicacion

- Debemos usar terminos simples y claros, evitando el tecnisismo, podemos mostrar los datos de manera visual, utilizando graficos claros y faciles de entender.
- Usar un lenguaje accesible y evadir el uso de palabras difíciles de entender.
- Proporcionar información adicional sobre los datos, las limitaciones del estudio y los posibles sesgos.