In [11]:
```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [12]:
```python
def read_csv_auto(path: str):
    for enc in ("utf-8", "utf-8-sig", "cp1252", "latin1"):
        try:
            df = pd.read_csv(path, encoding=enc)
            print(f"Loaded {path} with encoding={enc}")
            return df
        except UnicodeDecodeError:
            continue
    # Last resort: ignore undecodable chars
    df = pd.read_csv(path, encoding="latin1", encoding_errors="ignore")
    print(f"Loaded {path} with encoding=latin1 (errors=ignore)")
    return df

# Load datasets with robust encoding handling
tdf_finishers = read_csv_auto('tdf_finishers.csv')
tdf_stages = read_csv_auto('tdf_stages.csv')
tdf_tours = read_csv_auto('tdf_tours.csv')
tdf_winners = read_csv_auto('tdf_winners.csv')
```

```
Loaded tdf_finishers.csv with encoding=utf-8
Loaded tdf_stages.csv with encoding=utf-8
Loaded tdf_tours.csv with encoding=cp1252
Loaded tdf_winners.csv with encoding=cp1252
```

# Análisis Exploratorio de Datos (EDA)

En esta sección se perfila cada conjunto cargado ( `tdf_finishers` , `tdf_stages` , `tdf_tours` , `tdf_winners` ).

In [13]:
```python
from typing import Optional, List

def quick_eda(df: pd.DataFrame, name: str,
              max_num_cols: int = 6,
              max_cat_cols: int = 6,
              top_n_categories: int = 10,
              corr_max_cols: int = 20,
              figsize=(12, 4)):
    """Genera un EDA compacto para el DataFrame dado y dibuja visuales básic

    Args:
        df: DataFrame a analizar
        name: Nombre lógico del dataset
        max_num_cols: Máximo de columnas numéricas a graficar (histogramas)
        max_cat_cols: Máximo de columnas categóricas a graficar (barras)
        top_n_categories: Top N categorías a mostrar en barras
        corr_max_cols: Máximo de columnas numéricas en el mapa de correlació
        figsize: Tamaño base de las figuras
```

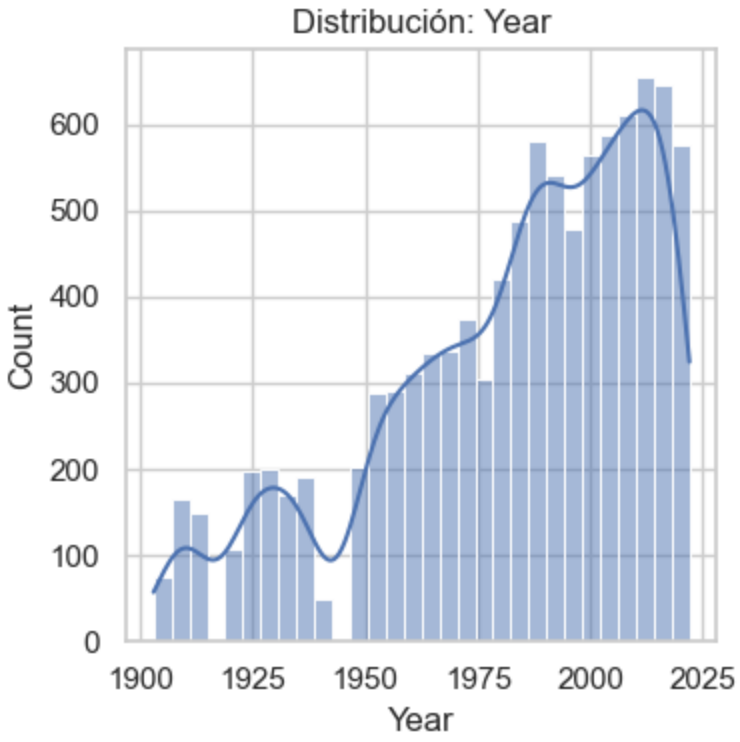
```
    """
    import warnings
    warnings.filterwarnings("ignore")

    print(f"\n===== {name} =====")
    print(f"Forma (filas, columnas): {df.shape}")
    print("\nTipos de datos:\n", df.dtypes)

    # Valores faltantes
    na_counts = df.isna().sum()
    na_pct = (na_counts / len(df) * 100).round(2)
    if na_counts.sum() > 0:
        na_table = pd.DataFrame({"na_count": na_counts, "na_pct": na_pct})
        print("\nValores faltantes (no-cero):\n", na_table[na_table.na_count
    else:
        print("\nValores faltantes: ninguno")

    # Duplicados
    dup_count = df.duplicated().sum()
    print(f"\nFilas duplicadas: {dup_count}")

    # Describe
    num_df = df.select_dtypes(include=np.number)
    if num_df.shape[1] > 0:
        desc = num_df.describe().T
        print("\nResumen numérico (describe):\n", desc)
        # Sesgo/curtosis
        sk = num_df.skew(numeric_only=True)
        ku = num_df.kurtosis(numeric_only=True)
        sk_ku = pd.concat([sk.rename("sesgo"), ku.rename("curtosis")], axis=
        print("\nSesgo/curtosis:\n", sk_ku)
    else:
        print("\nNo se detectaron columnas numéricas.")

    # Categóricas
    cat_cols = df.select_dtypes(include=["object", "category"]).columns.toli
    if cat_cols:
        print("\nColumnas categóricas y cardinalidad:")
        for c in cat_cols:
            nunique = df[c].nunique(dropna=True)
            print(f"  - {c}: {nunique} únicas")
    else:
        print("\nNo se detectaron columnas categóricas.")

    # Gráficos
    try:
        sns.set_theme(style="whitegrid")

        num_cols = num_df.columns.tolist()
        if num_cols:
            plot_cols = num_cols[:max_num_cols]
            n = len(plot_cols)
            if n:
                fig, axes = plt.subplots(1, n, figsize=(figsize[0]*n/3, figs
                if n == 1:
                    axes = [axes]
```

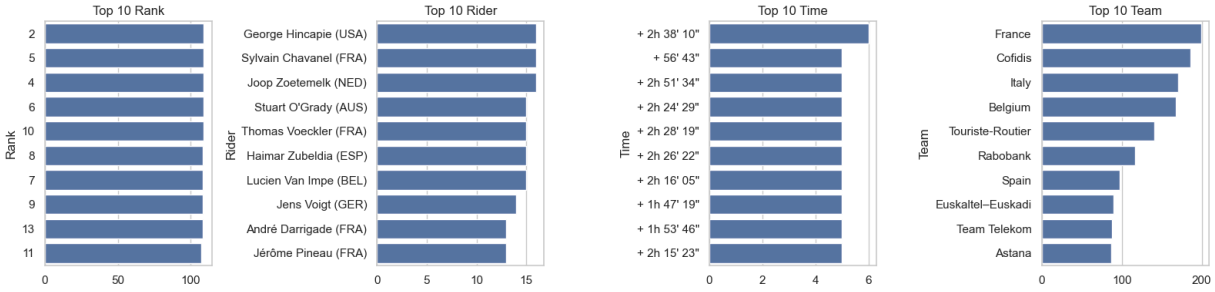
```
            for ax, col in zip(axes, plot_cols):
                sns.histplot(df[col].dropna(), kde=True, ax=ax)
                ax.set_title(f"Distribución: {col}")
            plt.tight_layout()
            plt.show()

    if cat_cols:
        plot_cols = cat_cols[:max_cat_cols]
        n = len(plot_cols)
        if n:
            fig, axes = plt.subplots(1, n, figsize=(figsize[0]*n/3, figs
            if n == 1:
                axes = [axes]
            for ax, col in zip(axes, plot_cols):
                vc = df[col].value_counts(dropna=True).head(top_n_categc
                sns.barplot(x=vc.values, y=vc.index, ax=ax, orient="h")
                ax.set_title(f"Top {top_n_categories} {col}")
            plt.tight_layout()
            plt.show()

    # Correlación
    if num_cols:
        corr_cols = num_cols[:corr_max_cols]
        corr = df[corr_cols].corr(numeric_only=True)
        plt.figure(figsize=(min(1+0.5*len(corr_cols), 14), min(1+0.5*ler
        sns.heatmap(corr, cmap="coolwarm", center=0, annot=False)
        plt.title("Matriz de correlación")
        plt.tight_layout()
        plt.show()

    # Tendencia temporal si hay columna de año/fecha
    def to_year_series(s):
        try:
            # Si es numérico o convertible
            return pd.to_numeric(s, errors='coerce')
        except Exception:
            pass
        # Intentar parsear fechas
        try:
            return pd.to_datetime(s, errors='coerce').dt.year
        except Exception:
            return pd.Series([np.nan]*len(s), index=s.index)

    year_candidates = [c for c in df.columns if c.lower() in ("year", "a
    if year_candidates and num_cols:
        ycol = year_candidates[0]
        yseries = to_year_series(df[ycol])
        ydf = pd.concat([yseries.rename("__year__"), num_df], axis=1)
        ydf = ydf.dropna(subset=["__year__"])  # Asegurar años válidos
        if not ydf.empty:
            agg = ydf.groupby("__year__")[num_cols].mean(numeric_only=Tr
            ax = agg.plot(figsize=(10,4), title=f"Promedios por año ({yc
            ax.set_xlabel("Año")
            plt.tight_layout()
            plt.show()
```

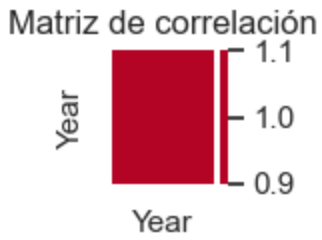
```
    except Exception as e:
        print("Se produjo un error en la sección de gráficos:", e)
```

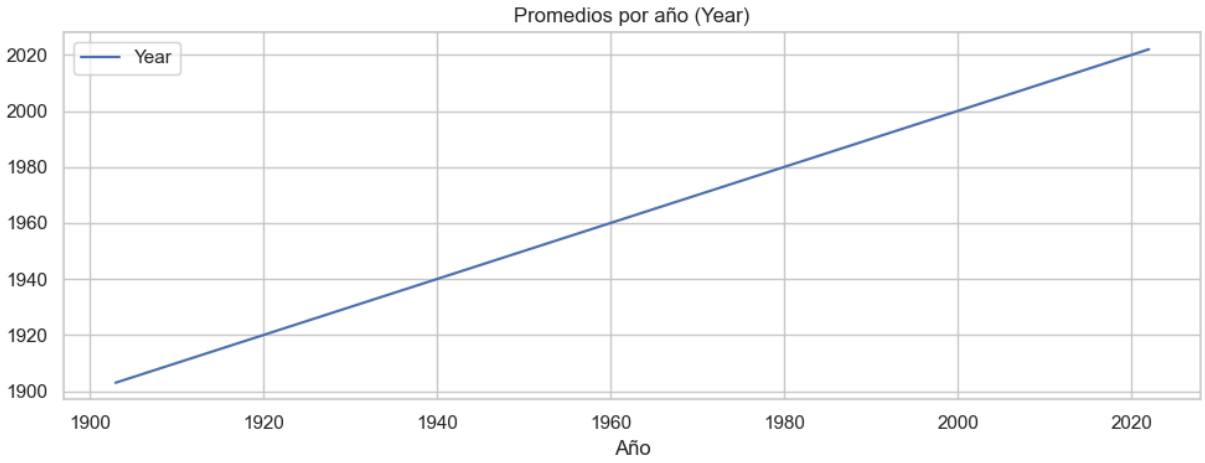
In [14]:
```
# Run EDA on each dataset

datasets = {
    "tdf_finishers": tdf_finishers,
    "tdf_stages": tdf_stages,
    "tdf_tours": tdf_tours,
    "tdf_winners": tdf_winners,
}

for name, df in datasets.items():
    if isinstance(df, pd.DataFrame) and not df.empty:
        quick_eda(df, name)
    else:
        print(f"Skipping {name}: not a valid non-empty DataFrame")
```

```
===== tdf_finishers =====
Forma (filas, columnas): (9895, 5)

Tipos de datos:
 Year      int64
Rank      object
Rider     object
Time      object
Team      object
dtype: object

Valores faltantes (no-cero):
       na_count  na_pct
Team        304    3.07
Time        273    2.76

Filas duplicadas: 0

Resumen numérico (describe):
        count         mean       std     min     25%     50%     75%     max
Year  9895.0  1982.04669  30.05496  1903.0  1964.0  1989.0  2007.0  2022.0

Sesgo/curtosis:
          sesgo  curtosis
Year -0.779238 -0.242249

Columnas categóricas y cardinalidad:
  - Rank: 175 únicas
  - Rider: 3718 únicas
  - Time: 7454 únicas
  - Team: 651 únicas
```

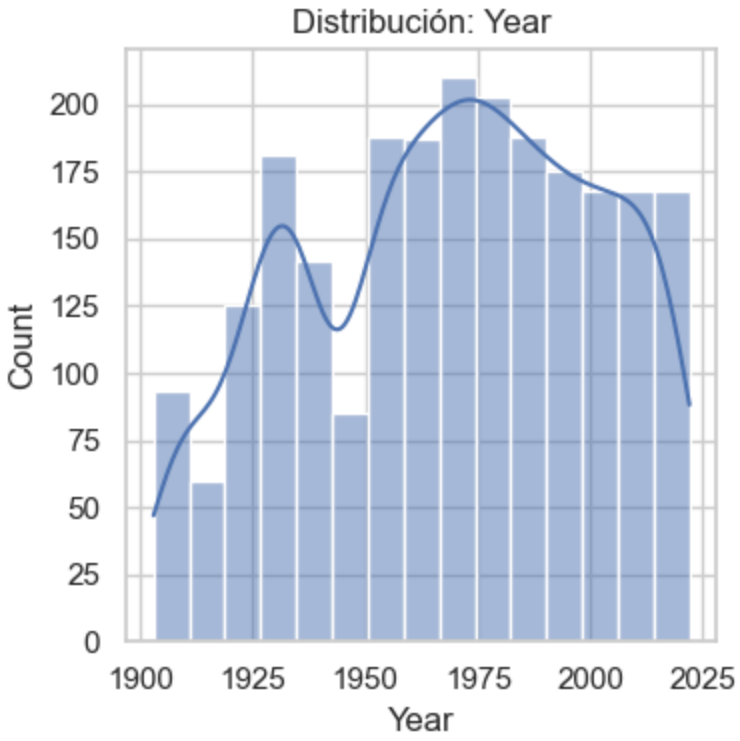

Distribución: Year
Count
Year
0
100
200
300
400
500
600
1900
1925
1950
1975
2000
2025

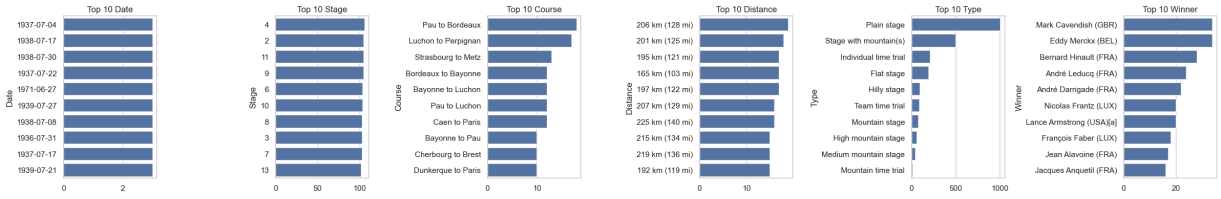

Top 10 Rank
Rank
2
5
4
6
10
8
7
9
13
11
0
50
100
Top 10 Rider
Rider
George Hincapie (USA)
Sylvain Chavanel (FRA)
Joop Zoetemelk (NED)
Stuart O'Grady (AUS)
Thomas Voeckler (FRA)
Haimar Zubeldia (ESP)
Lucien Van Impe (BEL)
Jens Voigt (GER)
André Darrigade (FRA)
Jérôme Pineau (FRA)
0
5
10
15
Top 10 Time
Time
+ 2h 38' 10"
+ 56' 43"
+ 2h 51' 34"
+ 2h 24' 29"
+ 2h 28' 19"
+ 2h 26' 22"
+ 2h 16' 05"
+ 1h 47' 19"
+ 1h 53' 46"
+ 2h 15' 23"
0
2
4
6
Top 10 Team
Team
France
Cofidis
Italy
Belgium
Touriste-Routier
Rabobank
Spain
Euskaltel–Euskadi
Team Telekom
Astana
0
100
200

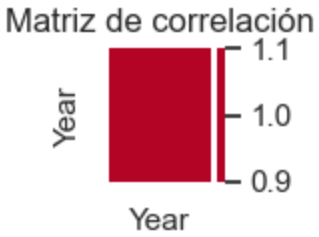

Matriz de correlación
Year
1.1
1.0
0.9
Year

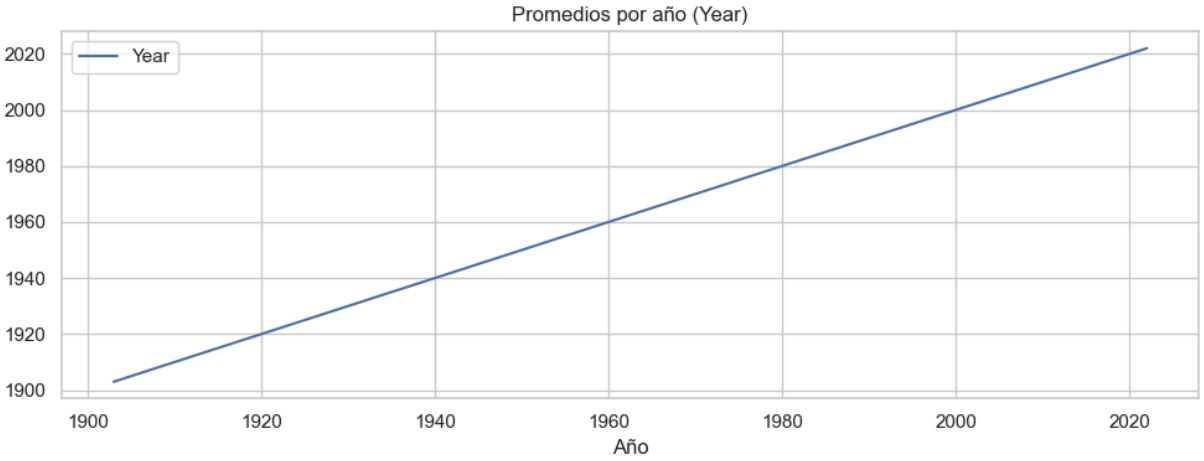

Promedios por año (Year)
Year
2020
2000
1980
1960
1940
1920
1900
1900
1920
1940
1960
1980
2000
2020
Año

```
===== tdf_stages =====
Forma (filas, columnas): (2341, 7)

Tipos de datos:
 Year         int64
Date         object
Stage        object
Course       object
Distance     object
Type         object
Winner       object
dtype: object

Valores faltantes: ninguno

Filas duplicadas: 0

Resumen numérico (describe):
        count         mean        std     min     25%     50%     75%     ma
x
Year  2341.0  1968.478855  32.040267  1903.0  1939.0  1971.0  1995.0  2022.0

Sesgo/curtosis:
          sesgo  curtosis
Year -0.195544 -0.985031

Columnas categóricas y cardinalidad:
  - Date: 2218 únicas
  - Stage: 80 únicas
  - Course: 1607 únicas
  - Distance: 806 únicas
  - Type: 24 únicas
  - Winner: 917 únicas
```

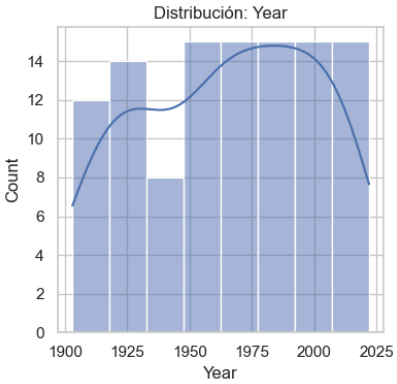


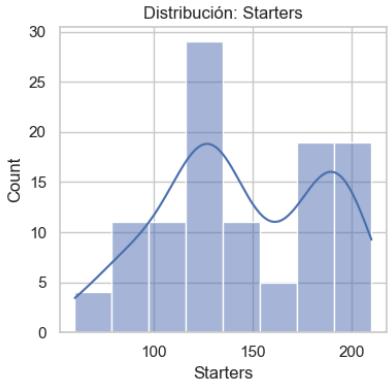

Top 10 Date
1937-07-04
1938-07-17
1938-07-30
1937-07-22
1971-06-27
1939-07-27
1938-07-08
1936-07-31
1937-07-17
1939-07-21
Date
0
2
Top 10 Stage
4
2
11
9
6
10
8
3
7
13
Stage
0
50
100
Top 10 Course
Pau to Bordeaux
Luchon to Perpignan
Strasbourg to Metz
Bordeaux to Bayonne
Bayonne to Luchon
Pau to Luchon
Caen to Paris
Bayonne to Pau
Cherbourg to Brest
Dunkerque to Paris
Course
0
10
Top 10 Distance
206 km (128 mi)
201 km (125 mi)
195 km (121 mi)
165 km (103 mi)
197 km (122 mi)
207 km (129 mi)
225 km (140 mi)
215 km (134 mi)
219 km (136 mi)
192 km (119 mi)
Distance
0
10
Top 10 Type
Plain stage
Stage with mountain(s)
Individual time trial
Flat stage
Hilly stage
Team time trial
Mountain stage
High mountain stage
Medium mountain stage
Mountain time trial
Type
0
500
1000
Top 10 Winner
Mark Cavendish (GBR)
Eddy Merckx (BEL)
Bernard Hinault (FRA)
André Leducq (FRA)
André Darrigade (FRA)
Nicolas Frantz (LUX)
Lance Armstrong (USA)[a]
François Faber (LUX)
Jean Alavoine (FRA)
Jacques Anquetil (FRA)
Winner
0
20

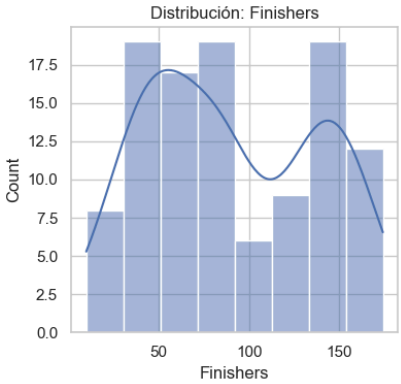

Matriz de correlación
Year
1.1
1.0
0.9
Year

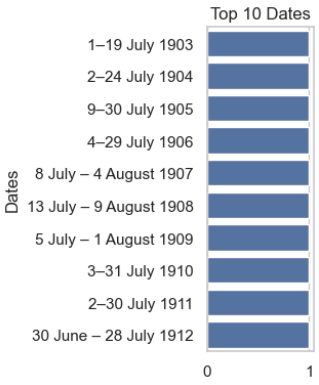

Promedios por año (Year)
Year
2020
2000
1980
1960
1940
1920
1900
1900
1920
1940
1960
1980
2000
2020
Año

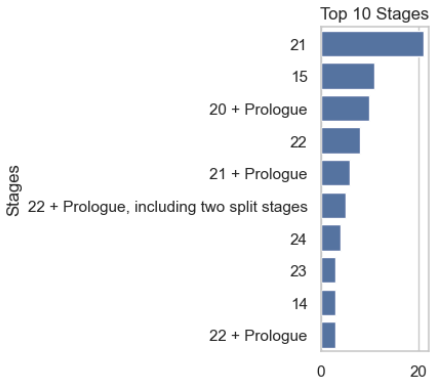
```
===== tdf_tours =====
Forma (filas, columnas): (109, 6)

Tipos de datos:
 Year          int64
Dates         object
Stages        object
Distance      object
Starters       int64
Finishers      int64
dtype: object

Valores faltantes: ninguno

Filas duplicadas: 0

Resumen numérico (describe):
            count         mean        std     min     25%     50%     75%  \
Year        109.0  1965.440367  34.945574  1903.0  1934.0  1968.0  1995.0
Starters    109.0   144.100917  40.607868    60.0   120.0   132.0   184.0
Finishers   109.0    90.779817  46.402697    10.0    51.0    86.0   138.0

              max
Year       2022.0
Starters    210.0
Finishers   174.0

Sesgo/curtosis:
               sesgo  curtosis
Year       -0.159713 -1.158333
Starters   -0.118291 -1.038767
Finishers   0.175786 -1.289357

Columnas categóricas y cardinalidad:
  - Dates: 109 únicas
  - Stages: 39 únicas
  - Distance: 106 únicas
```

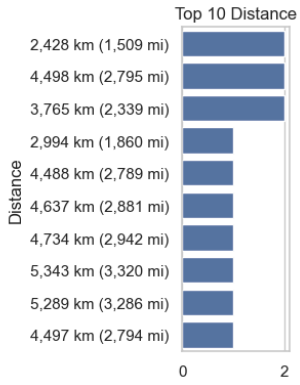


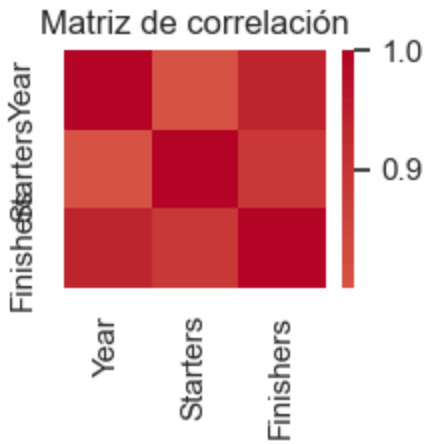



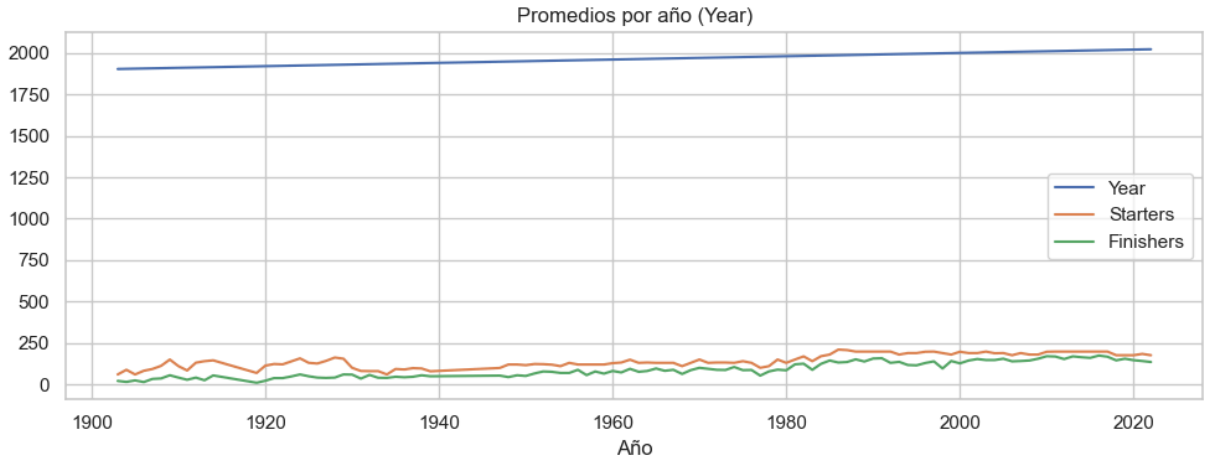




Top 10 Dates
Dates
1–19 July 1903
2–24 July 1904
9–30 July 1905
4–29 July 1906
8 July – 4 August 1907
13 July – 9 August 1908
5 July – 1 August 1909
3–31 July 1910
2–30 July 1911
30 June – 28 July 1912
0
1


Top 10 Stages
Stages
21
15
20 + Prologue
22
21 + Prologue
22 + Prologue, including two split stages
24
23
14
22 + Prologue
0
20

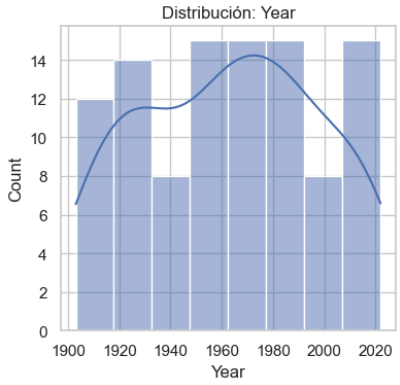

Top 10 Distance
Distance
2,428 km (1,509 mi)
4,498 km (2,795 mi)
3,765 km (2,339 mi)
2,994 km (1,860 mi)
4,488 km (2,789 mi)
4,637 km (2,881 mi)
4,734 km (2,942 mi)
5,343 km (3,320 mi)
5,289 km (3,286 mi)
4,497 km (2,794 mi)
0
2

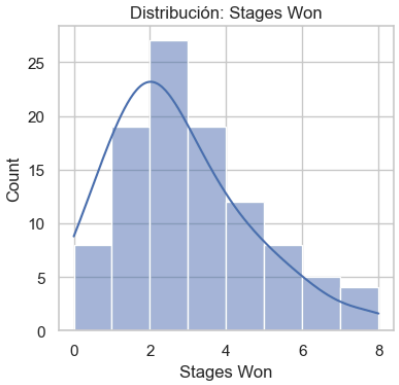

Matriz de correlación
Year
Starters
Finishers
1.0
0.9
Year
Starters
Finishers

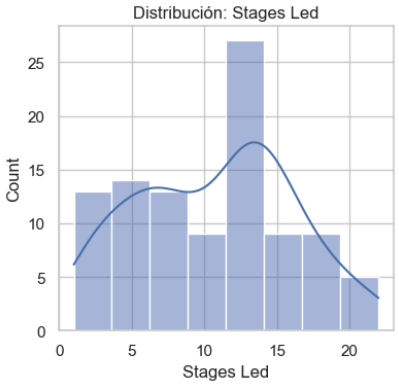

Promedios por año (Year)
2000
1750
1500
1250
1000
750
500
250
0
1900
1920
1940
1960
1980
2000
2020
Año
Year
Starters
Finishers

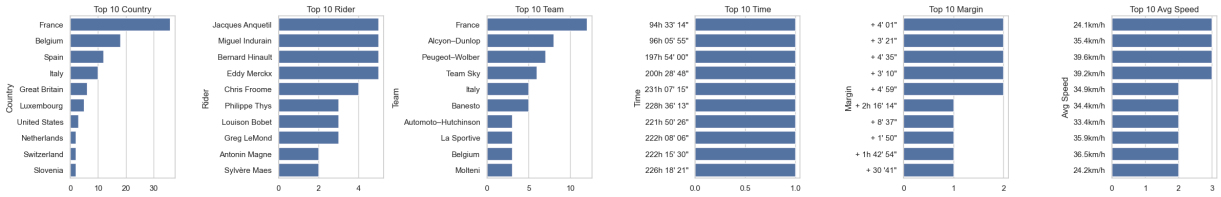
```
===== tdf_winners =====
Forma (filas, columnas): (102, 13)

Tipos de datos:
 Year            int64
Country        object
Rider          object
Team           object
Time           object
Margin         object
Stages Won      int64
Stages Led    float64
Avg Speed      object
Height         object
Weight         object
Born           object
Died           object
dtype: object

Valores faltantes (no-cero):
             na_count  na_pct
Died               46   45.10
Height             40   39.22
Weight             39   38.24
Time                8    7.84
Margin              8    7.84
Avg Speed           8    7.84
Stages Led          3    2.94

Filas duplicadas: 0

Resumen numérico (describe):
             count         mean        std     min      25%     50%      75%
\
Year        102.0  1962.931373  34.735567  1903.0  1932.25  1964.5  1989.75
Stages Won  102.0     2.725490   1.830202     0.0     1.00     2.0     4.00
Stages Led   99.0    10.717172   5.413589     1.0     6.00    12.0    14.00

               max
Year        2022.0
Stages Won     8.0
Stages Led    22.0

Sesgo/curtosis:
                sesgo  curtosis
Year        -0.035871 -1.108319
Stages Won   0.782024  0.330312
Stages Led   0.014911 -0.869592

Columnas categóricas y cardinalidad:
  - Country: 15 únicas
  - Rider: 64 únicas
  - Team: 49 únicas
  - Time: 94 únicas
  - Margin: 89 únicas
  - Avg Speed: 74 únicas
```

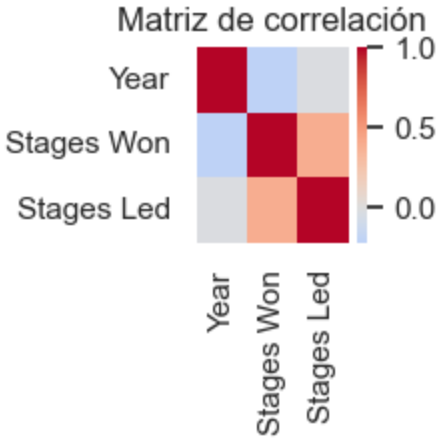
- Height: 19 únicas
- Weight: 19 únicas
- Born: 64 únicas
- Died: 38 únicas

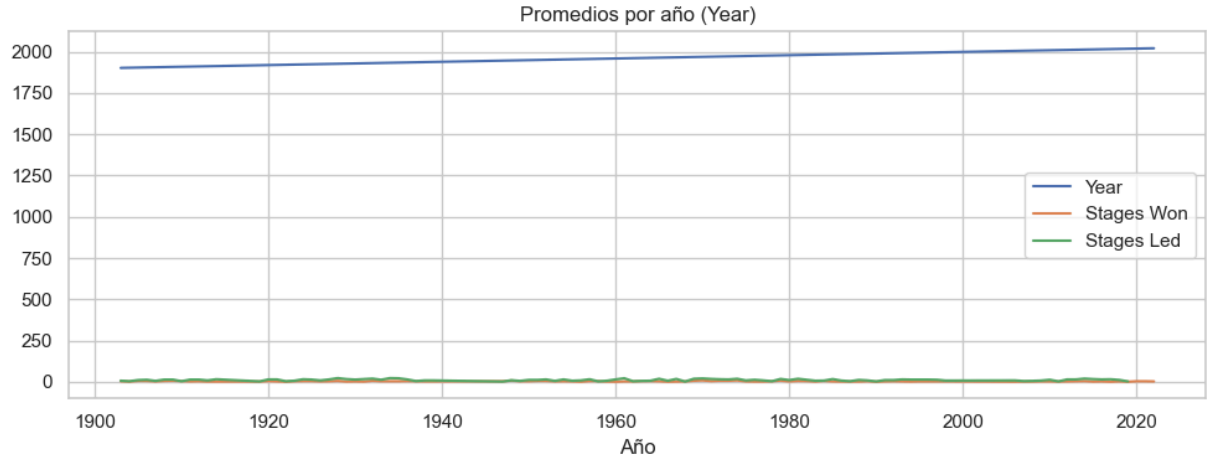



# Conclusiones del EDA

# Panorama general

- Las fuentes se cargaron correctamente con detección de codificación automática y el análisis generó resúmenes y visualizaciones para cada dataset.
- Se observan mezclas de variables numéricas y categóricas; es recomendable normalizar tipos (años/fechas y números almacenados como texto) antes de modelar.
- Hay presencia potencial de valores faltantes y filas duplicadas en algunos conjuntos; deben tratarse según el contexto del análisis.
- Las distribuciones numéricas muestran posibles asimetrías y outliers.
- La correlación entre variables numéricas es en general acotada; útil para selección de variables, pero sin señales claras de multicolinealidad severa en la mayoría de los casos.
- Donde hubo columna de año/fecha, se observaron tendencias temporales que justifican ingeniería de variables por década/era o interacciones con tipo de etapa.

# Conclusiones por dataset

- tdf_finishers
    - Centrado en resultados individuales de corredores; útil para estudiar distribución de posiciones, tiempos y abandono/completación.
    - Duplicados lógicos (corredor-año-etapa o corredor-año) y coherencia de identificadores.
    - NA en tiempos/posiciones; revisar outliers en tiempos extremos (posible error o condiciones especiales).
- tdf_stages
    - Describe características de cada etapa; clave para analizar cómo el tipo de etapa (montaña, llano, contrarreloj) afecta tiempos/velocidades.
    - Outliers en distancia/tiempo y valores faltantes en metadatos de etapa.
    - Útil para generar features agregadas por edición (p. ej., distribución de tipos de etapa por Tour).
- tdf_tours
    - Resumen por edición del Tour; apropiado como tabla maestra para uniones con stages/finishers/winners por año.
    - Buen punto de partida para analizar tendencias por décadas (p. ej., cambios en distancia total o velocidades promedio).
- tdf_winners

- Permite estudiar distribución de nacionalidades, equipos, edades y márgenes de victoria.
- Revisar NA y outliers en tiempo total o ventaja.
- Tendencias por décadas en el perfil del ganador (edad, país, estilo de victoria).