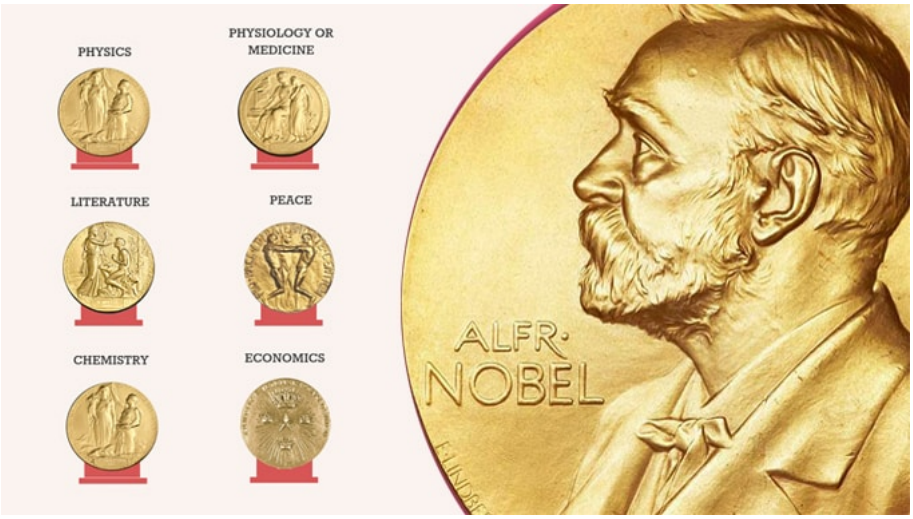
Analysis of Nobel Prize Winners (1901-2022)



- About Dataset
  - Introduction: The Nobel Prize is one of the most prestigious International honors awarded annually in recognition of outstanding achievements in various fields. Established in 1895 by the will of Alfred Nobel, a Swedish inventor, engineer, and industrialist, the Nobel Prize celebrates individuals and organizations that have made remarkable contributions to humanity in six categories: Physics, Chemistry, Physiology or Medicine, Literature, Peace, and Economic Sciences. The prizes are bestowed based on the recommendations of expert committees and organizations specific to each category. The Nobel Prizes have become a symbol of excellence and innovation, encouraging groundbreaking discoveries, literary excellence, peaceful diplomacy, and economic advancements. The laureates, from diverse backgrounds and disciplines, have shaped our world and improved the lives of countless individuals through their remarkable accomplishments. The Nobel Prize continues to inspire and recognize those who strive to make significant positive impacts on society and contribute to the advancement of knowledge and human welfare.
  - Content: This dataset includes records of all the Nobel Laureates (Individuals or Organizations) awarded till date.

Exploratory Data Analysis (EDA)-

Importing Necessary Libraries

In [1]:

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

```

In [3]:

```
%matplotlib inline
```

Understanding dataset -

In [186]:

```
data = pd.read_csv("D:\\Python EDA\\Nobel\\nobel_latest.csv")
```

In [187]:

```
1 data.head(5)
```

Out[187]:

	Year	Laureate_Id	Firstname	Lastname	Category	Gender	Prize_Share	Motivation	Birth_Date	Birth_Country	Birth_City	Birth_Country_Code	De
0	1901	1	Wilhelm Conrad	Röntgen	physics	male	1	"in recognition of the extraordinary services ...	1845-03-27	Prussia (now Germany)	Lennep (now Remscheid)	DE	1(
1	1901	293	Emil	von Behring	medicine	male	1	"for his work on serum therapy especially its ...	1854-03-15	Prussia (now Poland)	Hansdorf (now Lawice)	PL	31
2	1901	462	Henry	Dunant	peace	male	2	"for his humanitarian efforts to help wounded ...	1828-05-08	Switzerland	Geneva	CH	3(
3	1901	463	Frédéric	Passy	peace	male	2	"for his lifelong work for international peace...	1822-05-20	France	Paris	FR	12
4	1901	569	Sully	Prudhomme	literature	male	1	"in special recognition of his poetic composit...	1839-03-16	France	Paris	FR	07

In [188]:

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 989 entries, 0 to 988
Data columns (total 19 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   Year                  989 non-null    int64
 1   Laureate_Id           989 non-null    int64
 2   Firstname             989 non-null    object
 3   Lastname              957 non-null    object
 4   Category              989 non-null    object
 5   Gender                989 non-null    object
 6   Prize_Share           989 non-null    int64
 7   Motivation            989 non-null    object
 8   Birth_Date            989 non-null    object
 9   Birth_Country         958 non-null    object
 10  Birth_City            956 non-null    object
 11  Birth_Country_Code    958 non-null    object
 12  Death_Date            989 non-null    object
 13  Death_Country         646 non-null    object
 14  Death_City            640 non-null    object
 15  Death_Country_Code    646 non-null    object
 16  Organization_Name     727 non-null    object
 17  Organization_City     722 non-null    object
 18  Organization_Country  724 non-null    object
dtypes: int64(3), object(16)
memory usage: 146.9+ KB
```

In [189]:

```
1 data.isnull().sum()
```

Out[189]:

```
Year                     0
Laureate_Id              0
Firstname                0
Lastname                32
Category                 0
Gender                   0
Prize_Share              0
Motivation               0
Birth_Date               0
Birth_Country           31
Birth_City              33
Birth_Country_Code      31
Death_Date               0
Death_Country          343
Death_City             349
Death_Country_Code     343
Organization_Name      262
Organization_City      267
Organization_Country   265
dtype: int64
```

In [190]:

```
data.shape
```

Out[190]:

```
(989, 19)
```

Delete unnecessary columns-

In [191]:

```
df = data.drop(["Death_Country","Death_City","Death_Country_Code"],axis = 1)
```

In [192]:

```
df.head()
```

Out[192]:

	Year	Laureate_Id	Firstname	Lastname	Category	Gender	Prize_Share	Motivation	Birth_Date	Birth_Country	Birth_City	Birth_Country_Code	De
0	1901	1	Wilhelm Conrad	Röntgen	physics	male	1	"in recognition of the extraordinary services ...	1845-03-27	Prussia (now Germany)	Lennep (now Remscheid)	DE	1(
1	1901	293	Emil	von Behring	medicine	male	1	"for his work on serum therapy especially its ...	1854-03-15	Prussia (now Poland)	Hansdorf (now Lawice)	PL	31
2	1901	462	Henry	Dunant	peace	male	2	"for his humanitarian efforts to help wounded ...	1828-05-08	Switzerland	Geneva	CH	3(
3	1901	463	Frédéric	Passy	peace	male	2	"for his lifelong work for international peace...	1822-05-20	France	Paris	FR	12
4	1901	569	Sully	Prudhomme	literature	male	1	"in special recognition of his poetic composit...	1839-03-16	France	Paris	FR	07

Handling null values -

In [193]:

```
df.isnull().sum()
```

Out[193]:

```
Year                    0
Laureate_Id             0
Firstname               0
Lastname               32
Category                0
Gender                  0
Prize_Share             0
Motivation              0
Birth_Date              0
Birth_Country          31
Birth_City             33
Birth_Country_Code     31
Death_Date              0
Organization_Name     262
Organization_City     267
Organization_Country  265
dtype: int64
```

- Handling them one by one .

In [194]:

```
df["Birth_Country"].fillna("Not mentioned",inplace = True)
```

In [195]:

```
df["Birth_City"].fillna("Not mentioned",inplace = True)
```

In [196]:

```
df["Birth_Country_Code"].fillna("Not mentioned",inplace = True)
```

In [330]:

```
df["Organization_Name"].fillna("Not mentioned",inplace = True)
```

In [198]:

```
df["Organization_City"].fillna("Not mentioned",inplace = True)
```

In [199]:

```
df["Organization_Country"].fillna("Not mentioned",inplace = True)
```

In [200]:

```
df.isnull().sum()
```

Out[200]:

```
Year                    0
Laureate_Id             0
Firstname               0
Lastname               32
Category                0
Gender                  0
Prize_Share             0
Motivation              0
Birth_Date              0
Birth_Country           0
Birth_City              0
Birth_Country_Code      0
Death_Date              0
Organization_Name       0
Organization_City       0
Organization_Country    0
dtype: int64
```

Column - concat()

first_name ,last_name

In [204]:

```
df["Firstname"].unique()
```

Out[204]:

```
array(['Wilhelm Conrad', 'Emil', 'Henry', 'Frédéric', 'Sully',
       'Jacobus H.', 'Theodor', 'Albert', 'Élie', 'Pieter', 'Ronald',
       'Hendrik A.', 'Pierre', 'Randal', 'Henri', 'Marie', 'Niels Ryberg',
       'Bjørnstjerne', 'Svante', 'José', 'Lord',
       'Institute of International Law', 'Sir William', 'Ivan', 'Adolf',
       'Henryk', 'Philipp', 'Bertha', 'Robert', 'Giosuè', 'Theodore',
       'J.J.', 'Santiago', 'Camillo', 'Alphonse', 'Eduard', 'Rudyard',
       'Louis', 'Ernesto Teodoro', 'Albert A.', 'Ernest', 'Rudolf',
       'Fredrik', 'Ilya', 'Paul', 'Klas Pontus', 'Gabriel', 'Guglielmo',
       'Selma', 'Wilhelm', 'Ferdinand', 'Auguste', 'Paul Henri',
       'Permanent International Peace Bureau', 'Johannes Diderik',
       'Albrecht', 'Otto', 'Alfred', 'Tobias', 'Maurice', 'Allvar',
       'Gerhart', 'Alexis', 'Victor', 'Gustaf', 'Elihu', 'Charles',
       'Rabindranath', 'Heike', 'Max', 'Theodore W.', 'William',
       'Lawrence', 'Richard', 'Romain', 'Verner',
       'International Committee of the Red Cross', 'Charles Glover',
       'Karl', 'Henrik', 'Fritz', 'Woodrow', 'Jules', 'Johannes', 'Carl',
       'Charles Edouard', 'Walther', 'August', 'Léon', 'Knut', 'Anatole',
```

In [20]:

```
df["Lastname"].unique()
```

Out[20]:

```
array(['Röntgen', 'von Behring', 'Dunant', 'Passy', 'Prudhomme',
       "van 't Hoff", 'Fischer', 'Mommsen', 'Gobat', 'Ducommun', 'Zeeman',
       'Ross', 'Lorentz', 'Curie', 'Cremer', 'Becquerel', 'Finsen',
       'Bjørnson', 'Arrhenius', 'Echegaray', 'Rayleigh', nan, 'Ramsay',
       'Pavlov', 'Mistral', 'von Baeyer', 'Sienkiewicz', 'Lenard',
       'von Suttner', 'Koch', 'Carducci', 'Roosevelt', 'Thomson',
       'Ramón y Cajal', 'Moissan', 'Golgi', 'Laveran', 'Buchner',
       'Kipling', 'Renault', 'Moneta', 'Michelson', 'Rutherford',
       'Eucken', 'Bajer', 'Mechnikov', 'Ehrlich', 'Arnoldson', 'Lippmann',
       'Marconi', 'Kocher', 'Lagerlöf', 'Ostwald', 'Braun', 'Beernaert',
       "d'Estournelles de Constant", 'Heyse', 'van der Waals', 'Kossel',
       'Wallach', 'Fried', 'Asser', 'Wien', 'Maeterlinck', 'Gullstrand',
       'Hauptmann', 'Carrel', 'Sabatier', 'Grignard', 'Dalén', 'Root',
       'Richet', 'Werner', 'La Fontaine', 'Tagore', 'Kamerlingh Onnes',
       'von Laue', 'Bárány', 'Richards', 'Bragg', 'Willstätter',
       'Rolland', 'von Heidenstam', 'Barkla', 'Gjellerup', 'Pontoppidan',
       'Planck', 'Haber', 'Wilson', 'Bordet', 'Stark', 'Spitteler',
       'Guillaume', 'Nernst', 'Krogh', 'Bourgeois', 'Hamsun', 'France',
```

In [205]:

```
df["Lastname"].replace(np.nan," ",inplace = True)
```

In [206]:

```
df.columns
```

Out[206]:

```
Index(['Year', 'Laureate_Id', 'Firstname', 'Lastname', 'Category', 'Gender',
       'Prize_Share', 'Motivation', 'Birth_Date', 'Birth_Country',
       'Birth_City', 'Birth_Country_Code', 'Death_Date', 'Organization_Name',
       'Organization_City', 'Organization_Country'],
      dtype='object')
```

In [207]:

```
df["fullname"] = df["Firstname"]+" "+df["Lastname"]

```

In [208]:

```
df.drop(["Firstname","Lastname"],axis = 1,inplace = True)
```

In [209]:

```
df.isnull().sum()
```

Out[209]:

```
Year                    0
Laureate_Id             0
Category                0
Gender                  0
Prize_Share             0
Motivation              0
Birth_Date              0
Birth_Country           0
Birth_City              0
Birth_Country_Code      0
Death_Date              0
Organization_Name       0
Organization_City       0
Organization_Country    0
fullname                0
dtype: int64
```

Handling columns ony by one -

Column - "Category"

In [211]:

```
df["Category"].unique()
```

Out[211]:

```
array(['physics', 'medicine', 'peace', 'literature', 'chemistry',
       'economics'], dtype=object)
```

Column - "Gender"

In [213]:

```
1 df["Gender"].unique()
```

Out[213]:

```
array(['male', 'female', 'org'], dtype=object)
```

In [214]:

```
1 df[df["Gender"]=="org"] # org are the institute
```

237	1947	508	peace	org	2	pioneering work in the internationa...	1647-00-00	Not mentioned	Not mentioned	Not mentioned	0000-00-00	Not mentioned	N
238	1947	509	peace	org	2	"for their pioneering work in the internationa...	1917-00-00	Not mentioned	Not mentioned	Not mentioned	0000-00-00	Not mentioned	N
285	1954	515	peace	org	1	"for its efforts to heal the wounds of war by ...	14-12-1950	Not mentioned	Not mentioned	Not mentioned	0000-00-00	Not mentioned	N
347	1963	482	peace	org	2	"for promoting the principles of the Geneva Co...	1863-00-00	Not mentioned	Not mentioned	Not mentioned	0000-00-00	Not mentioned	N
348	1963	523	peace	org	2	"for promoting the principles of the Geneva	1919-00-00	Not mentioned	Not mentioned	Not mentioned	0000-00-00	Not mentioned	N

Column - "Prize_Share"

In [215]:

```
1 df.columns
```

Out[215]:

```
Index(['Year', 'Laureate_Id', 'Category', 'Gender', 'Prize_Share',
       'Motivation', 'Birth_Date', 'Birth_Country', 'Birth_City',
       'Birth_Country_Code', 'Death_Date', 'Organization_Name',
       'Organization_City', 'Organization_Country', 'fullname'],
      dtype='object')
```

In [216]:

```
1 df["Prize_Share"].unique()
```

Out[216]:

```
array([1, 2, 4, 3], dtype=int64)
```

Column - "Birth_Date"

In [221]:

```
1 df["Birth_Date"].unique()
```

```
       '18-05-1951', '09-10-1948', '10-08-1951', '24-05-1942',
       '24-05-1941', '22-06-1943', '14-09-1951', '21-09-1934',
       '09-02-1945', '21-10-1944', '2007-00-00', '08-11-1954',
       '19-07-1945', '12-09-1940', '08-06-1942', '01-06-1940',
       '12-09-1945', '29-09-1932', '28-03-1949', '07-03-1944',
       '03-05-1945', '29-01-1962', '1955-00-00', '1993-00-00',
       '01-03-1955', '14-04-1951', '10-03-1941', '25-07-1956',
       '22-06-1944', '02-09-1922', '27-01-1942', '07-08-1948',
       '27-05-1959', '30-01-1948', '21-02-1961', '12-11-1964',
       '06-12-1942', '25-10-1972', '22-12-1941', '15-08-1976',
       '23-02-1966', '12-01-1942', '25-04-1935', '12-07-1956',
       '14-05-1954', '23-11-1957', '25-07-1922', '16-05-1937',
       '20-04-1948', '22-04-1943', '19-02-1964', '11-12-1968',
       '24-03-1952', '08-08-1931', '25-08-1952', '1949-00-00',
       '12-09-1935', '16-06-1965', '18-09-1960', '1956-00-00',
       '03-09-1963', '02-10-1963', '16-03-1968', '21-09-1931',
       '04-08-1948', '25-10-1931', '1967-00-00', '04-11-1955',
       '11-01-1968', '13-12-1953', '1987-00-00', '25-09-1962',
       '01-09-1940', '1954-00-00', '20-05-1945', '01-12-1942',
       '15-06-1947', '20-04-1955', '25-10-1953', '10-10-1966'
```

- Here many elements have different format of data eg - (25-04-1900, 1890-12-21,1987-00-00)

In [222]:

```
#Extracting Year
def extract_birth_year(x):
    y = x.split("-")
    if len(y[0]) == 2:
        return int(y[2])
    elif len(y[0]) == 4:
        return int(y[0])
df["Birth_Year"] = df["Birth_Date"].apply(extract_birth_year)
```

In [223]:

```
df['Birth_Year'].unique()
```

Out[223]:

```
array([1845, 1854, 1828, 1822, 1839, 1852, 1817, 1843, 1833, 1865, 1857,
       1853, 1859, 1867, 1860, 1832, 1842, 1873, 1849, 1830, 1835, 1846,
       1862, 1858, 1856, 1871, 1837, 1844, 1874, 1841, 1850, 1829, 1891,
       1847, 1864, 1838, 1869, 1866, 1861, 1879, 1876, 1868, 1890, 1872,
       1863, 1877, 1870, 1851, 1885, 1884, 1886, 1882, 1887, 1878, 1892,
       1875, 1881, 1888, 1883, 1889, 1901, 1902, 1893, 1900, 1897, 1905,
       1921, 1895, 1903, 1898, 1906, 1904, 1896, 1647, 1917, 1899, 1880,
       1907, 1912, 1914, 1910, 1915, 1916, 1950, 1911, 1913, 1908, 1922,
       1926, 1918, 1909, 1925, 1920, 1929, 1928, 1919, 1946, 1927, 1930,
       1931, 1940, 1923, 1924, 1934, 1938, 1944, 1936, 1943, 1961, 1933,
       1894, 1932, 1937, 1980, 1941, 1947, 1939, 1948, 1935, 1942, 1945,
       1959, 1957, 1949, 1992, 1971, 1951, 1956, 1976, 1952, 1960, 1988,
       1953, 1955, 1974, 1958, 1979, 1972, 1967, 1969, 1962, 1997, 1954,
       1963, 2013, 2007, 1993, 1964, 1966, 1968, 1965, 1987], dtype=int64)
```

Column - "Birth_Country"

In [180]:

```
1 df["Birth_Country"].unique()
```

Out[180]:

```
array(['Prussia (now Germany)', 'Prussia (now Poland)', 'Switzerland',
       'France', 'the Netherlands', 'Schleswig (now Germany)', 'India',
       'United Kingdom', 'Russian Empire (now Poland)',
       'Faroe Islands (Denmark)', 'Norway', 'Sweden', 'Spain',
       'Not mentioned', 'Scotland', 'Russia', 'Poland',
       'Hungary (now Slovakia)', 'Austrian Empire (now Czech Republic)',
       'Germany', 'Tuscany (now Italy)', 'USA', 'Italy',
       'Bavaria (now Germany)', 'British India (now India)',
       'Austrian Empire (now Italy)', 'New Zealand',
       'East Friesland (now Germany)', 'Denmark',
       'Russian Empire (now Ukraine)', 'Luxembourg',
       'Russian Empire (now Latvia)', 'Hesse-Kassel (now Germany)',
       'Belgium', 'Mecklenburg (now Germany)', 'Germany (now Russia)',
       'Austria', 'Prussia (now Russia)', 'Australia', 'Canada',
       'Austria-Hungary (now Slovenia)', 'Ireland',
       'Java Dutch East Indies (now Indonesia)',
       'Austrian Empire (now Austria)', 'Germany (now Poland)',
       'Württemberg (now Germany)', 'Argentina',
       'Austria-Hungary (now Hungary)', 'Austria-Hungary (now Austria)',
       'Austria-Hungary (now Croatia)', 'Russian Empire (now Finland)',
       'Austria-Hungary (now Poland)', 'Chile',
       'Austria-Hungary (now Czech Republic)', 'Portugal', 'Japan',
       'South Africa', 'Germany (now France)', 'Iceland', 'China',
       'French Algeria (now Algeria)', 'Brazil', 'Guadeloupe Island',
       'Southern Rhodesia (now Zimbabwe)', 'Hungary',
       'Bosnia (now Bosnia and Herzegovina)',
       'Russian Empire (now Azerbaijan)', 'Ottoman Empire (now Turkey)',
       'USSR (now Russia)', 'Egypt', 'Austria-Hungary (now Ukraine)',
       'Guatemala', 'Russian Empire (now Belarus)', 'Vietnam', 'Romania',
       'Austria-Hungary (now Bosnia and Herzegovina)',
       'Russian Empire (now Russia)', 'Northern Ireland',
       'Poland (now Lithuania)', 'British West Indies (now Saint Lucia)',
       'Ottoman Empire (now North Macedonia)', 'Crete (now Greece)',
       'India (now Pakistan)', 'Venezuela',
       'Russian Empire (now Lithuania)', 'Bulgaria',
       'Poland (now Ukraine)', 'Colombia', 'Lithuania', 'Mexico',
       'German-occupied Poland (now Poland)', 'Madagascar',
       'West Germany (now Germany)', 'Taiwan', 'Nigeria',
       'Korea (now South Korea)', 'Costa Rica', 'Tibet (now China)',
       'Burma (now Myanmar)', 'Saint Lucia',
       'British Mandate of Palestine (now Israel)',
       'Poland (now Belarus)', 'East Timor',
       'Free City of Danzig (now Poland)', 'USSR (now Belarus)',
       'Gold Coast (now Ghana)', 'Trinidad and Tobago', 'Iran', 'Kenya',
       'British Protectorate of Palestine (now Israel)',
       'British India (now Bangladesh)', 'Turkey', 'Persia (now Iran)',
       'Czechoslovakia (now Czech Republic)', 'Finland', 'Peru', 'Cyprus',
       'Yemen', 'Liberia', 'Morocco', 'Pakistan', 'Ukraine', 'Iraq',
       'Belgian Congo (now Democratic Republic of the Congo)', 'Ethiopia',
       'Philippines', 'Lebanon'], dtype=object)
```

In [ ]:

```
1
```

In [ ]:

```
1 #Step One
```

In [229]:

```
def extracting_text(string):
    if "(" in string:
        start_index = string.index("(")
        end_index = string.index(")")
        extracted_text = string[start_index + 1 : end_index]
        extracted_text = extracted_text.strip()
        return extracted_text
    elif string =="nan":
        return "not specified"
    else :
        return string
df["country"] = df["Birth_Country"].apply(extracting_text)
df["country"].unique()
```

Out[229]:

```
array(['now Germany', 'now Poland', 'Switzerland', 'France',
       'the Netherlands', 'India', 'United Kingdom', 'Denmark', 'Norway',
       'Sweden', 'Spain', 'Not mentioned', 'Scotland', 'Russia', 'Poland',
       'now Slovakia', 'now Czech Republic', 'Germany', 'now Italy',
       'USA', 'Italy', 'now India', 'New Zealand', 'now Ukraine',
       'Luxembourg', 'now Latvia', 'Belgium', 'now Russia', 'Austria',
       'Australia', 'Canada', 'now Slovenia', 'Ireland', 'now Indonesia',
       'now Austria', 'Argentina', 'now Hungary', 'now Croatia',
       'now Finland', 'Chile', 'Portugal', 'Japan', 'South Africa',
       'now France', 'Iceland', 'China', 'now Algeria', 'Brazil',
       'Guadeloupe Island', 'now Zimbabwe', 'Hungary',
       'now Bosnia and Herzegovina', 'now Azerbaijan', 'now Turkey',
       'Egypt', 'Guatemala', 'now Belarus', 'Vietnam', 'Romania',
       'Northern Ireland', 'now Lithuania', 'now Saint Lucia',
       'now North Macedonia', 'now Greece', 'now Pakistan', 'Venezuela',
       'Bulgaria', 'Colombia', 'Lithuania', 'Mexico', 'Madagascar',
       'Taiwan', 'Nigeria', 'now South Korea', 'Costa Rica', 'now China',
       'now Myanmar', 'Saint Lucia', 'now Israel', 'East Timor',
       'now Ghana', 'Trinidad and Tobago', 'Iran', 'Kenya',
       'now Bangladesh', 'Turkey', 'now Iran', 'Finland', 'Peru',
       'Cyprus', 'Yemen', 'Liberia', 'Morocco', 'Pakistan', 'Ukraine',
       'Iraq', 'now Democratic Republic of the Congo', 'Ethiopia',
       'Philippines', 'Lebanon'], dtype=object)
```

In [230]:

```
# Step two
def handling_now(value):
    if "now" in value:
        return value.replace("now","")
    else :
        return  value
df["country"] = df["country"].apply(handling_now)
df["country"].unique()
```

Out[230]:

```
array([' Germany', ' Poland', 'Switzerland', 'France', 'the Netherlands',
       'India', 'United Kingdom', 'Denmark', 'Norway', 'Sweden', 'Spain',
       'Not mentioned', 'Scotland', 'Russia', 'Poland', ' Slovakia',
       ' Czech Republic', 'Germany', ' Italy', 'USA', 'Italy', ' India',
       'New Zealand', ' Ukraine', 'Luxembourg', ' Latvia', 'Belgium',
       ' Russia', 'Austria', 'Australia', 'Canada', ' Slovenia',
       'Ireland', ' Indonesia', ' Austria', 'Argentina', ' Hungary',
       ' Croatia', ' Finland', 'Chile', 'Portugal', 'Japan',
       'South Africa', ' France', 'Iceland', 'China', ' Algeria',
       'Brazil', 'Guadeloupe Island', ' Zimbabwe', 'Hungary',
       ' Bosnia and Herzegovina', ' Azerbaijan', ' Turkey', 'Egypt',
       'Guatemala', ' Belarus', 'Vietnam', 'Romania', 'Northern Ireland',
       ' Lithuania', ' Saint Lucia', ' North Macedonia', ' Greece',
       ' Pakistan', 'Venezuela', 'Bulgaria', 'Colombia', 'Lithuania',
       'Mexico', 'Madagascar', 'Taiwan', 'Nigeria', ' South Korea',
       'Costa Rica', ' China', ' Myanmar', 'Saint Lucia', ' Israel',
       'East Timor', ' Ghana', 'Trinidad and Tobago', 'Iran', 'Kenya',
       ' Bangladesh', 'Turkey', ' Iran', 'Finland', 'Peru', 'Cyprus',
       'Yemen', 'Liberia', 'Morocco', 'Pakistan', 'Ukraine', 'Iraq',
       ' Democratic Republic of the Congo', 'Ethiopia', 'Philippines',
       'Lebanon'], dtype=object)
```

In [232]:

```
# Step Three -
def handling_space(value):
    if " " in value:
        return value.replace(" ","")
    else :
        return  value
df["country"] = df["country"].apply(handling_space)
df["country"].unique()
```

Out[232]:

```
array(['Germany', 'Poland', 'Switzerland', 'France', 'theNetherlands',
       'India', 'UnitedKingdom', 'Denmark', 'Norway', 'Sweden', 'Spain',
       'Notmentioned', 'Scotland', 'Russia', 'Slovakia', 'CzechRepublic',
       'Italy', 'USA', 'NewZealand', 'Ukraine', 'Luxembourg', 'Latvia',
       'Belgium', 'Austria', 'Australia', 'Canada', 'Slovenia', 'Ireland',
       'Indonesia', 'Argentina', 'Hungary', 'Croatia', 'Finland', 'Chile',
       'Portugal', 'Japan', 'SouthAfrica', 'Iceland', 'China', 'Algeria',
       'Brazil', 'GuadeloupeIsland', 'Zimbabwe', 'BosniaandHerzegovina',
       'Azerbaijan', 'Turkey', 'Egypt', 'Guatemala', 'Belarus', 'Vietnam',
       'Romania', 'NorthernIreland', 'Lithuania', 'SaintLucia',
       'NorthMacedonia', 'Greece', 'Pakistan', 'Venezuela', 'Bulgaria',
       'Colombia', 'Mexico', 'Madagascar', 'Taiwan', 'Nigeria',
       'SouthKorea', 'CostaRica', 'Myanmar', 'Israel', 'EastTimor',
       'Ghana', 'TrinidadandTobago', 'Iran', 'Kenya', 'Bangladesh',
       'Peru', 'Cyprus', 'Yemen', 'Liberia', 'Morocco', 'Iraq',
       'DemocraticRepublicoftheCongo', 'Ethiopia', 'Philippines',
       'Lebanon'], dtype=object)
```

Column -"Death_Date"

In [253]:

```
mask = (df["Death_Date"] == "0000-00-00")
df.loc[mask, "Death_Date"] = "Unknown"
```

In [255]:

```
df["Death_Date"].unique()
```

Out[255]:

```
array(['10-02-1923', '31-03-1917', '30-10-1910', '12-06-1912',
       '07-09-1907', '01-03-1911', '15-07-1919', '01-11-1903',
       '16-03-1914', '07-12-1906', '09-10-1943', '16-09-1932',
       '04-02-1928', '19-04-1906', '22-07-1908', '25-08-1908',
       '04-07-1934', '24-09-1904', '26-04-1910', '02-10-1927',
       '04-09-1916', '30-06-1919', 'Unknown', '23-07-1916', '27-02-1936',
       '25-03-1914', '20-08-1917', '15-11-1916', '20-05-1947',
       '21-06-1914', '27-05-1910', '16-02-1907', '06-01-1919',
       '30-08-1940', '17-10-1934', '20-02-1907', '21-01-1926',
       '18-05-1922', '13-08-1917', '18-01-1936', '08-02-1918',
       '10-02-1918', '09-05-1931', '19-10-1937', '14-09-1926',
       '22-01-1922', '15-07-1916', '20-08-1915', '20-02-1916',
       '13-07-1921', '20-07-1937', '27-07-1917', '16-03-1940',
       '04-04-1932', '20-04-1918', '06-10-1912', '15-05-1924',
       '02-04-1914', '08-03-1923', '05-07-1927', '26-02-1931',
       '04-05-1921', '29-07-1913', '30-08-1928', '06-05-1949',
       '28-07-1930', '06-06-1946', '05-11-1944', '14-08-1941',
       '13-12-1935', '09-12-1937', '07-02-1937', '04-12-1935',
```

In [259]:

```
def handling_death_date(value):
    if value == "Unknown":
        return 0
    else:
        value = value.split("-")
        return int(value[2])
df["Death_year"] =df["Death_Date"].apply(handling_death_date)
df["Death_year"].unique()


```

Out[259]:

```
array([1923, 1917, 1910, 1912, 1907, 1911, 1919, 1903, 1914, 1906, 1943,
       1932, 1928, 1908, 1934, 1904, 1927, 1916,    0, 1936, 1947, 1940,
       1926, 1922, 1918, 1931, 1937, 1915, 1921, 1924, 1913, 1949, 1930,
       1946, 1944, 1941, 1935, 1960, 1942, 1971, 1961, 1957, 1938, 1925,
       1952, 1955, 1956, 1962, 1945, 1951, 1954, 1977, 1953, 1939, 1978,
       1964, 1975, 1929, 1950, 1959, 1987, 1970, 1976, 1933, 1984, 1967,
       1981, 1958, 1974, 1966, 1968, 1991, 1986, 1973, 1995, 1969, 1965,
       1963, 1988, 1979, 1982, 1996, 1972, 1999, 1983, 1994, 2002, 1997,
       1985, 2008, 2003, 1998, 1993, 1989, 1992, 2013, 1990, 2007, 2006,
       1980, 2011, 2004, 2012, 2001, 2015, 2000, 2019, 2005, 2010, 2009,
       2014, 2017, 2021, 2022, 2018, 2020, 2023, 2016], dtype=int64)
```

Column - "Win_Age"

In [239]:

```
#creating age column
```

In [264]:

```
df["Win_Age"] = df["Year"]-df["Birth_Year"]
```

In [265]:

```
df["Win_Age"].unique()
```

Out[265]:

```
array([ 56,  47,  73,  79,  62,  49,  50,  85,  59,  69,  37,  45,  44,
        75,  51,  36,  43,  71,  72,  31,  52,  55,  74,  70,  48,  54,
        63,  42,  64,  35,  68,  80,  57,  19,  39,  58,  41,  67,  60,
        38,  46,  53,  25,  40,  61,  77,  32,  86,  65,  66,  17,  81,
        78, 300,  30,   4,  33,  34, 100,  87,  76,  16,  84,  82,  83,
         5,  28,  88,  90,  89,   2,  10,  96,  97,  15], dtype=int64)
```

In [275]:

```
def handling_win_age(value):
    if value <17:
        return np.nan
    if value > 96:
        return np.nan
    else :
        return value
df["Win_Age"] =df["Win_Age"].apply(handling_win_age)
df["Win_Age"].unique()
```

Out[275]:

```
array([56., 47., 73., 79., 62., 49., 50., 85., 59., 69., 37., 45., 44.,
       75., 51., 36., 43., 71., 72., 31., 52., 55., 74., 70., 48., 54.,
       63., 42., 64., 35., 68., 80., 57., 19., 39., 58., 41., 67., 60.,
       38., 46., 53., 25., 40., 61., 77., 32., 86., 65., 66., 17., 81.,
       78., nan, 30., 33., 34., 87., 76., 84., 82., 83., 28., 88., 90.,
       89., 96.])
```

In [278]:

```
df.columns
```

Out[278]:

```
Index(['Year', 'Laureate_Id', 'Category', 'Gender', 'Prize_Share',
       'Motivation', 'Birth_Country', 'Birth_City', 'Birth_Country_Code',
       'Death_Date', 'Organization_Name', 'Organization_City',
       'Organization_Country', 'fullname', 'Birth_Year', 'country',
       'Death_year', 'Win_Age'],
      dtype='object')
```

Column - "Laureate_Type"

In [285]:

```
# creating a laureate_type column
```

In [284]:

```
def handlinglaureate(gender):
    if gender =="org":
        return "Organisation"
    else :
        return "Individual"
df["Laureate_type"] = df["Gender"].apply(handlinglaureate)
df["Laureate_type"].unique()
```

Out[284]:

```
array(['Individual', 'Organisation'], dtype=object)
```

- Rearranging Columns -

In [288]:

```
columns = ['Year', 'Category', 'Laureate_Id', 'Laureate_type', 'fullname', 'Gender',
       'Prize_Share', 'Birth_Year','Win_Age', 'Birth_Country',
       'Birth_City', 'Birth_Country_Code', 'Death_Date',
       'Organization_Name','Organization_City', 'Organization_Country', 'Motivation']
df1 = df[columns]
```

In [289]:

```
df1.head()
```

Out[289]:

	Year	Category	Laureate_Id	Laureate_type	fullname	Gender	Prize_Share	Birth_Year	Win_Age	Birth_Country	Birth_City	Birth_Country_Code	D
0	1901	physics	1	Individual	Wilhelm Conrad Röntgen	male	1	1845	56.0	Prussia (now Germany)	Lennep (now Remscheid)	DE	1
1	1901	medicine	293	Individual	Emil von Behring	male	1	1854	47.0	Prussia (now Poland)	Hansdorf (now Lawice)	PL	3
2	1901	peace	462	Individual	Henry Dunant	male	2	1828	73.0	Switzerland	Geneva	CH	3
3	1901	peace	463	Individual	Frédéric Passy	male	2	1822	79.0	France	Paris	FR	1
4	1901	literature	569	Individual	Sully Prudhomme	male	1	1839	62.0	France	Paris	FR	0

In [ ]:

```

```

Visualization -

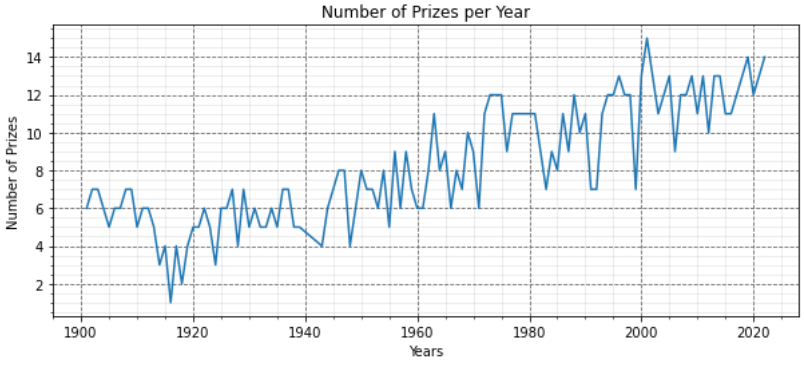
Analyzing the number of Nobel Prizes given over the years

In [290]:

```
temp_df = df["Year"].value_counts().sort_index().reset_index()
```

In [291]:

```
plt.figure(figsize = (10,4))

sns.lineplot(data=temp_df, x="index", y="Year")

plt.xlabel("Years")
plt.ylabel("Number of Prizes")

plt.title("Number of Prizes per Year")

plt.grid(visible=True, which='major', color='#666666', linestyle='--')

plt.minorticks_on()
plt.grid(visible=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
```



In [292]:

```
df1 = df.groupby("Year")["Laureate_Id"].count().cumsum().reset_index()
df1
```

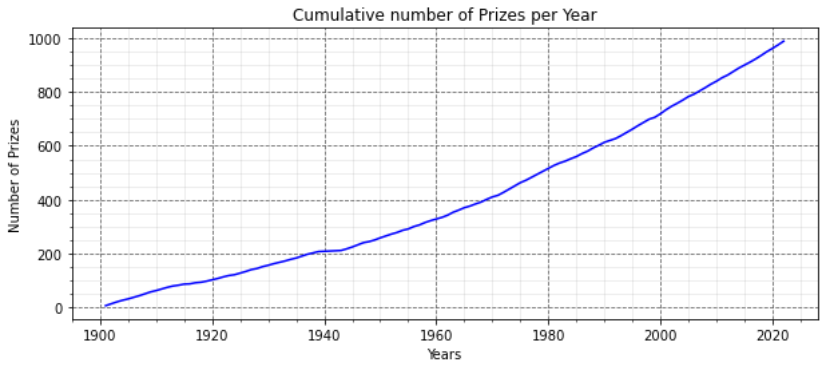
Out[292]:

	Year	Laureate_Id
0	1901	6
1	1902	13
2	1903	20
3	1904	26
4	1905	31
...	...	...
114	2018	936
115	2019	950
116	2020	962
117	2021	975
118	2022	989

119 rows × 2 columns

In [293]:

```
plt.figure(figsize = (10,4))

plt.plot(df1["Year"], df1["Laureate_Id"], color="blue")
plt.xlabel("Years")
plt.ylabel("Number of Prizes")

plt.title("Cumulative number of Prizes per Year")

plt.grid(visible=True, which='major', color='#666666', linestyle='--')

plt.minorticks_on()
plt.grid(visible=True, which='minor', color='#999999', linestyle='-', alpha=0.2)

plt.show()
```



Analyzing the Categories

In [294]:

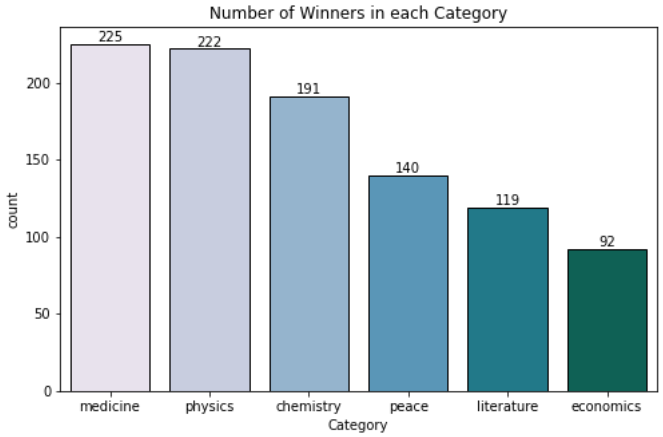
```
df["Category"].value_counts()
```

Out[294]:

```
medicine      225
physics       222
chemistry     191
peace         140
literature    119
economics      92
Name: Category, dtype: int64
```

In [297]:

```
igure(figsize = (8,5))

sns.countplot(data=df, x="Category", order=df["Category"].value_counts().index, palette = sns.color_palette("PuBuGn"), edgecolor='black')

abel in ax.containers:
x.bar_label(label)

itle("Number of Winners in each Category")
how()
```

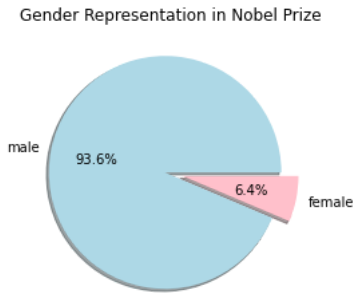


- We can observe that the Most number of Nobel Prizes are given in the Fields of Medicine and Physics
- Nobel Prizes in Economics were awarded from the year 1969. Hence the total awardees in this category are low.

Gender Representation in Nobel Prize.¶

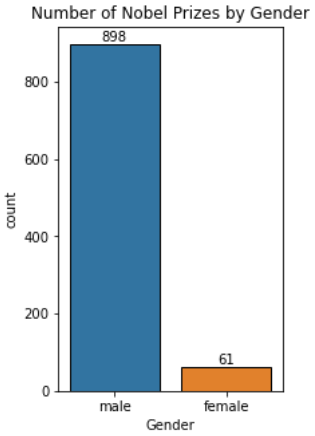
In [304]:

```
# Removing 'org' from our gender column

temp_df = df[df["Gender"]!="org"]

# Plotting a pie chart to analyze the percentage of male and female laureates.
freq = temp_df["Gender"].value_counts()
keys = freq.keys().to_list()
counts = freq.to_list()

color = ["lightblue", "pink"]
explode = [0.05, 0.1]

plt.pie(x=counts, labels=keys, autopct="%1.1f%%", colors = color, explode = explode, shadow=True)
plt.title("Gender Representation in Nobel Prize")
plt.show()
```



In [306]:

```
# Plotting a countplot to check the number of male and female laureates
plt.figure(figsize=(3,5))
ax = sns.countplot(data=temp_df, x="Gender", edgecolor='black')

for label in ax.containers:
    ax.bar_label(label)

plt.title("Number of Nobel Prizes by Gender")
plt.show()
```

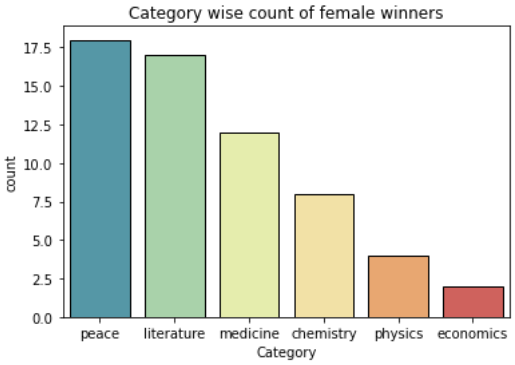


In [307]:

```
# Further analysis of female laureates
df_female = df[df["Gender"]=="female"]
```

In [310]:

```
# Plotting a countplot to find out the categories in which "female" laureates get the prize.
ax = sns.countplot(data=df_female, x="Category", order=df_female["Category"].value_counts().index, palette=sns.color_palette("Spectra

plt.title("Category wise count of female winners")
plt.show()
```

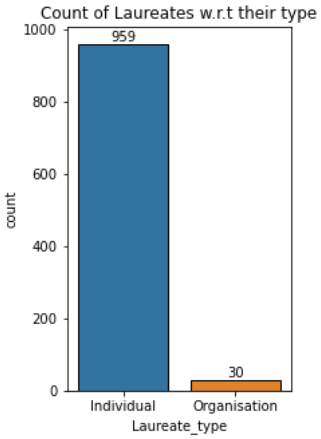


- We can observe that there is a Huge gap between the number of Male and Female Laureates.
- Out of the 959 Individual Laureates, there are 898 male laureates(93.6%) and only 61 female laureates(6.4%).
- Female laureates have mostly won Nobel Prizes in the categories of "Peace" and "Literature".

Analyisis of Laureate types

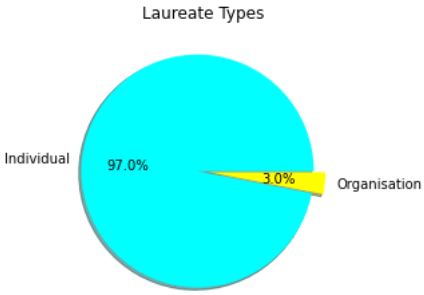
In [312]:

```
#Individuals or Organizations
plt.figure(figsize=(3,5))
ax = sns.countplot(data=df, x="Laureate_type", edgecolor='black')
for label in ax.containers:
    ax.bar_label(label)
plt.title("Count of Laureates w.r.t their type")
plt.show()
```



In [313]:

```
freq = df["Laureate_type"].value_counts()
keys = freq.keys().to_list()
counts = freq.to_list()

color = ["aqua", "yellow"]
explode = [0.05, 0.05]

plt.pie(x=counts, labels=keys, autopct="%1.1f%%", colors = color, explode = explode, shadow=True)
plt.title("Laureate Types")
plt.show()
```



In [317]:

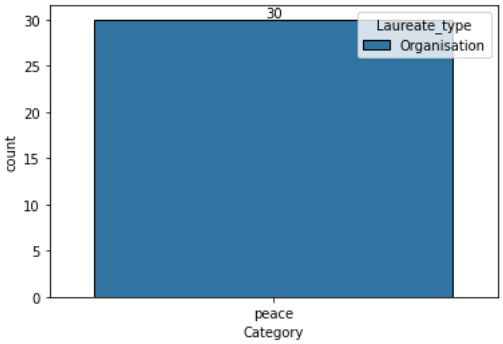
```
df_org = df[df["Laureate_type"] == "Organisation"]
df_org["Category"].value_counts()

```

Out[317]:

```
peace    30
Name: Category, dtype: int64
```

In [318]:

```
ax = sns.countplot(data=df_org, x="Category", hue="Laureate_type", order=df_org["Category"].value_counts().index, edgecolor='black')
for label in ax.containers:
    ax.bar_label(label)
plt.show()
```



- We can observe that a large majority of the Nobel Prize Winners are Individuals (97%).
- All the Organizations that have won the Nobel Prize, have won it in the category of "Peace"

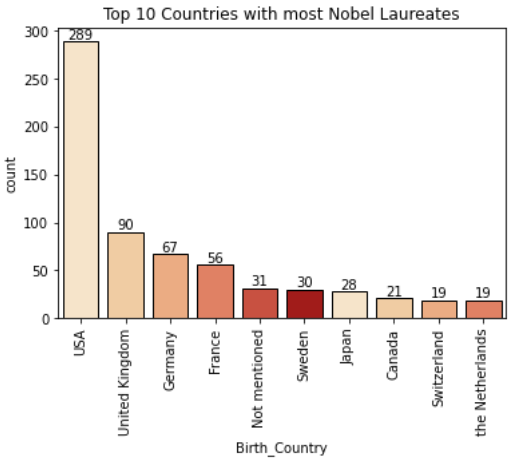
Nationality Analysis-

In [319]:

```
#Analyzing the birth country of nobel laureates
```

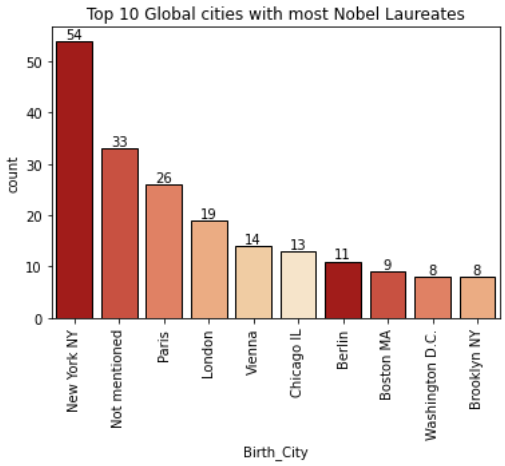
In [372]:

```
ot to show the top 10 countries that have produced the most number of Nobel Laureates.
ata=df, x="Birth_Country", order=df["Birth_Country"].value_counts().head(10).index, palette=sns.color_palette("OrRd"), edgecolor='black')
ainers:
el)

90)
untries with most Nobel Laureates")
```



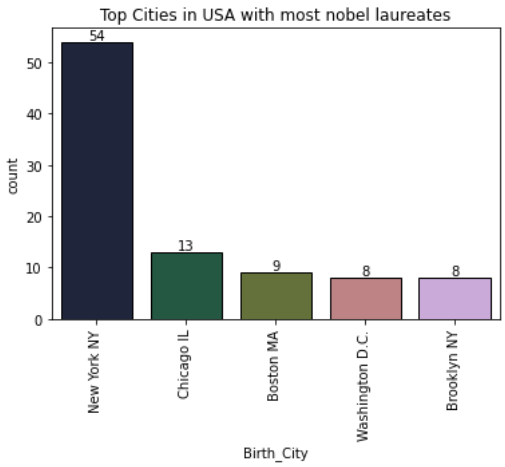
In [371]:

```
ntplot to showcase the top 10 cities that have produced the most nobel laureates.
ot(data=df, x="Birth_City", order=df["Birth_City"].value_counts().head(10).index, palette=sns.color_palette("OrRd_r"), edgecolor='black')
containers:
(label)

ion=90)
0 Global cities with most Nobel Laureates")
```



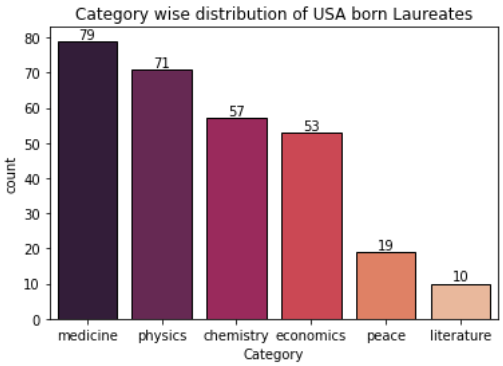
In [368]:

```
aureates born in USA
ntry"]=="USA"]
f_usa, x="Birth_City", order=df_usa["Birth_City"].value_counts().head().index, palette=sns.color_palette("cubehelix"), edgecolor='black')
s:


USA with most nobel laureates")
```



In [363]:

```
untplot(data=df_usa, x="Category", order=df_usa["Category"].value_counts().index, palette=sns.color_palette("rocket"), edgecolor='black')
n ax.containers:
label(label)
Category wise distribution of USA born Laureates")
```



- We can observe that USA has produced the lion's share of nobel laureates.
- On analyzing the cities that produced the most nobel laureates, we observe that most laureates were born in New York(USA).
- Top 5 cities in USA that produced most laureates include NewYork, Chicago, Boston, Washington DC and Brooklyn.
- On further analyzing the USA born laureates, we can observe that most of them have won the nobel prizes in the category of "Medicine", closely followed by the category of "Physics".

Analysis of Affiliated Organizations.

In [325]:

```
# Filtering the dataset to avoid the data with unknown values for affiliated organizations.
df_aff = df[df["Organization_Name"] != "None"]
df_aff.shape
# Observe that we have 727 laureates that had affiliations with some organizations
```

Out[325]:

```
(989, 19)
```

In [328]:

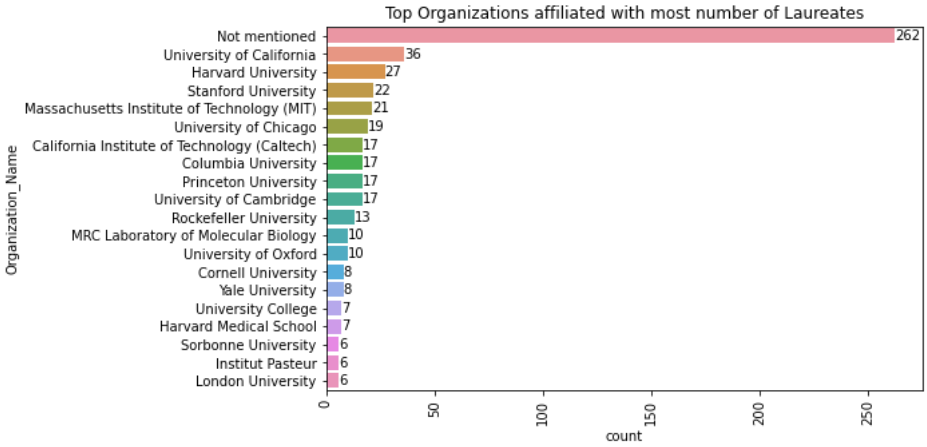
```
df_aff["Organization_Name"].value_counts().head(10)
```

Out[328]:

```
Not mentioned                                 262
University of California                       36
Harvard University                             27
Stanford University                            22
Massachusetts Institute of Technology (MIT)    21
University of Chicago                          19
California Institute of Technology (Caltech)   17
Columbia University                            17
Princeton University                           17
University of Cambridge                        17
Name: Organization_Name, dtype: int64
```

In [331]:

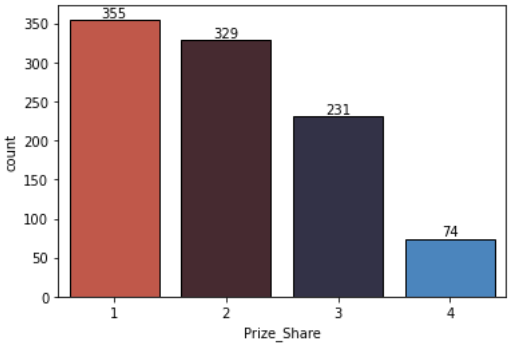
```
plt.figure(figsize=(8,5))
ax = sns.countplot(data=df_aff, y="Organization_Name", order=df_aff["Organization_Name"].value_counts().head(20).index)
for label in ax.containers:
    ax.bar_label(label)
plt.xticks(rotation=90)
plt.title("Top Organizations affiliated with most number of Laureates")
plt.show()
```



- We can observe that "University of California", "Harvard University", "Stanford University" and "MIT" are the top organizations with which Nobel Laureates are affiliated.

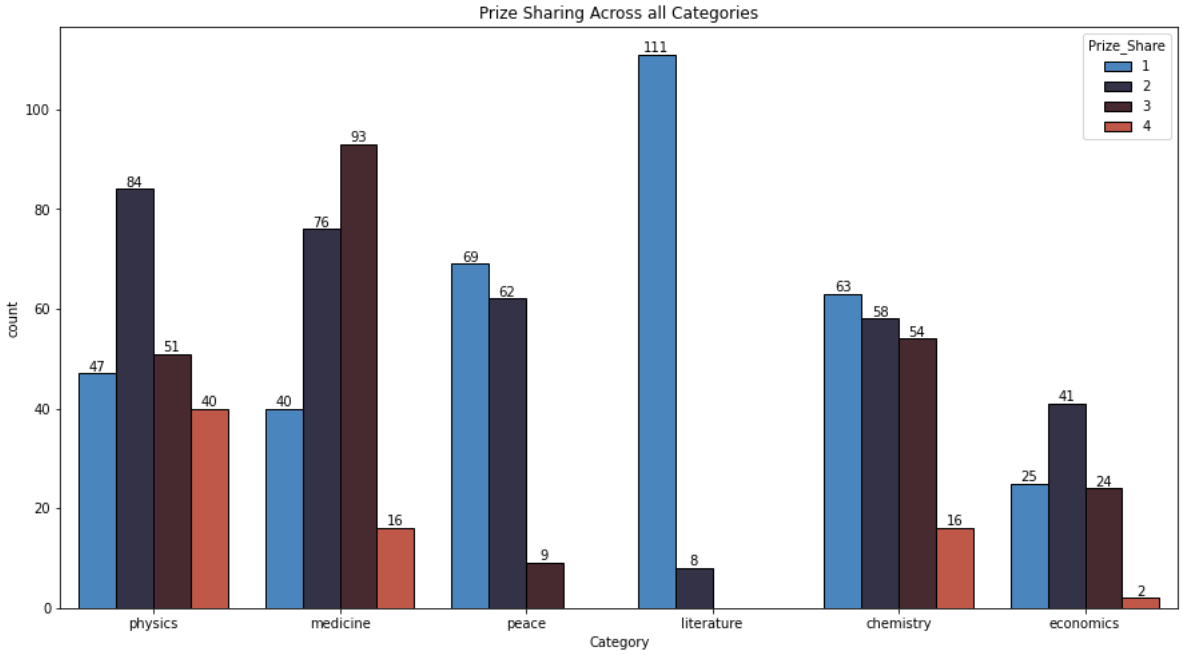
In [362]:

```
#The Nobel Prize could be split among up to 4 people who had contributed to the final solution.
ax = sns.countplot(data=df, x="Prize_Share", palette=sns.color_palette("icefire_r",4), edgecolor='black')
for label in ax.containers:
    ax.bar_label(label)
plt.show()
```



In [361]:

```
plt.figure(figsize = (15,8))
ax = sns.countplot(data=df, x="Category", hue="Prize_Share", palette = sns.color_palette("icefire", 4), edgecolor='black')
for label in ax.containers:
    ax.bar_label(label)
plt.title("Prize Sharing Across all Categories")
plt.show()
```



- Upon analyzing the above plots we can observe that number of nobel prizes won by Solo laureates is still the greatest.
- Considering that low hanging fruit is usually picked first, is it possible that the prize sharing has increased in contemporary times due to problems becoming harder and harder to the point where one person alone can't discover a solution completely on their own.
- Upon further analyzing the prize sharing in all categories, we can observe that:
- In the field of Physics, most nobel prizes are won by 2 individuals working together.
- In the field of Medicine, most nobel prizes are won by 3 individuals working together.
- In the field of Peace, most nobel prizes are won by Solo individuals, closely followed by 2 individuals working together.
- In the field of Literature, Individual creativity reigns supereme and most nobels are won by Solo individuals.
- In the field of Chemistry, most nobels are won by Solo laureates, but no. of prizes shared between 2 and 3 people is also very close.
- In the field of Economics, most prizes are won by 2 individuals working together

Analyzing Age of Laureates when they won the Nobel Prize.

In [337]:

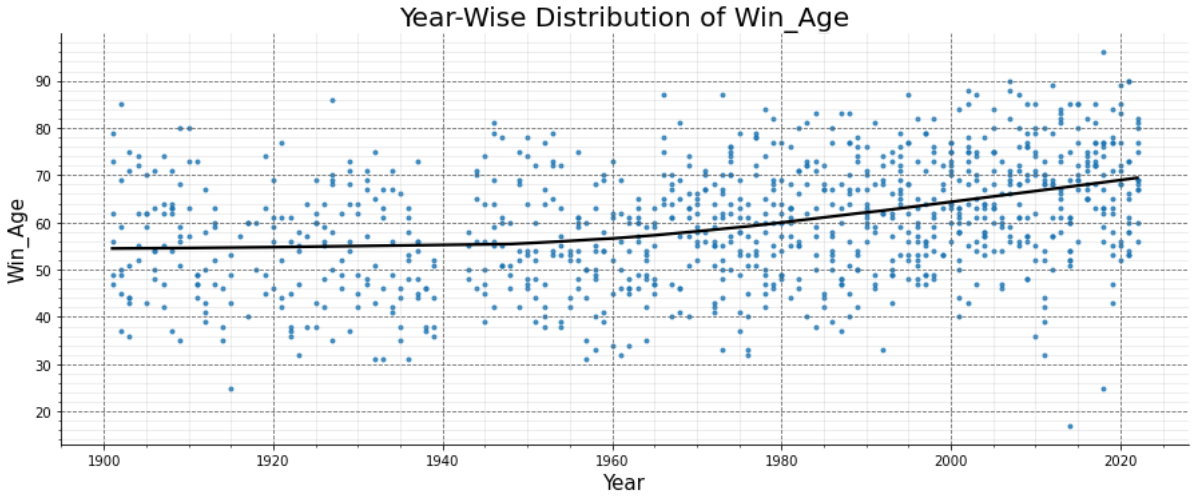
```
df_ind = df[df["Laureate_type"]=="Individual"]
# Finding the Overall mean age of Laureates.
mean_win_age = round(df_ind["Win_Age"].mean(), 2)
mean_win_age
```

Out[337]:

```
60.13
```

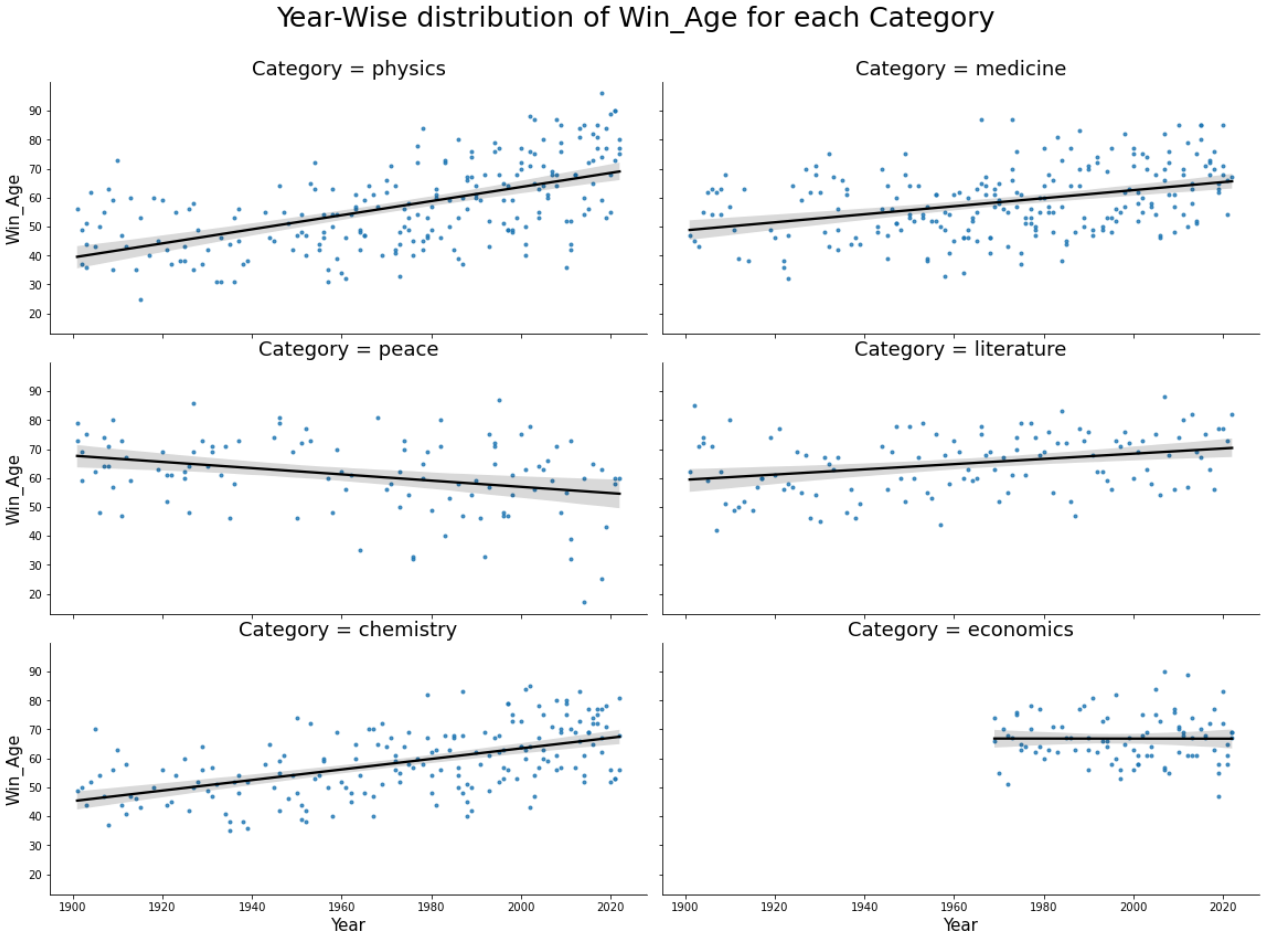
In [338]:

```
# Plotting a lmplot to showcase year-wise distribution of Win_Age of laureates
g=sns.lmplot(data=df_ind, x="Year", y="Win_Age", aspect=2.5, lowess=True, markers=".", line_kws={'color' : 'black'})
g.set_axis_labels("Year", "Win_Age", fontsize=15)
plt.title("Year-Wise Distribution of Win_Age", fontdict={'fontsize':20})
plt.grid(visible=True, which='major', color='#666666', linestyle='--')
plt.minorticks_on()
plt.grid(visible=True, which='minor', color='#999999', linestyle='-', alpha=0.2)
plt.show()
```



In [339]:

```
# Further the above plot for all the categories
g = sns.lmplot(data=df_ind, x="Year", y="Win_Age", col="Category", markers=".", col_wrap=2, height=4, aspect=2, line_kws={'color':'bl
g.set_titles(size=18)
g.set_axis_labels("Year", "Win_Age", fontsize=15)
plt.suptitle('Year-Wise distribution of Win_Age for each Category', fontsize=25)
plt.subplots_adjust(top=0.9)
plt.show()

```



In [340]:

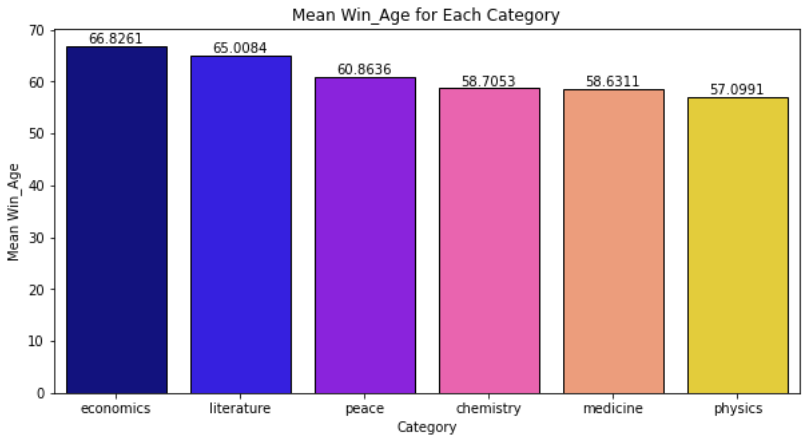
```
# Calculating mean Win_Age for all the different categories
temp_df = df_ind.groupby("Category")["Win_Age"].mean().sort_values(ascending=False).reset_index()
temp_df
```

Out[340]:

	Category	Win_Age
0	economics	66.826087
1	literature	65.008403
2	peace	60.863636
3	chemistry	58.705263
4	medicine	58.631111
5	physics	57.099099

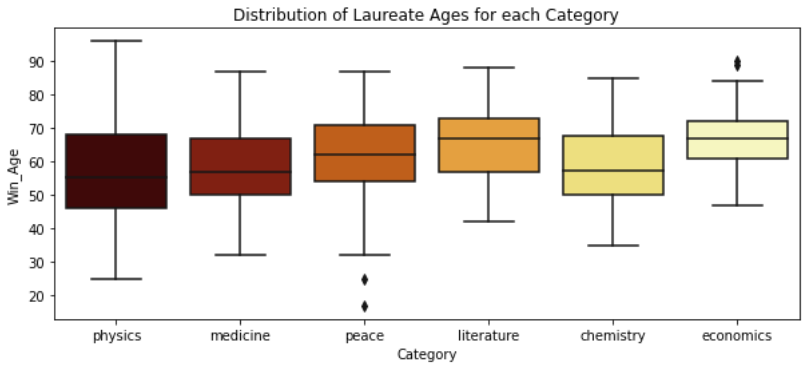
In [359]:

```
ing the Mean Age for all Categories
ure(figsize=(10,5))
s.barplot(data=temp_df, x='Category', y="Win_Age", palette=sns.color_palette(palette="gnuplot2", n_colors=6, desat=1), edgecolor='black')
el in ax.containers:
bar_label(label)

bel("Mean Win_Age")
le("Mean Win_Age for Each Category")
w()
```



In [358]:

```
plt.figure(figsize=(10,4))
sns.boxplot(data=df_ind, x="Category", y="Win_Age", palette=sns.color_palette(palette="afmhot", n_colors=6, desat=1))
plt.title("Distribution of Laureate Ages for each Category")
plt.show()

```



- The Average Win_Age of the Nobel laureate is growing over time, from early 50-ies in 1901 to late 60-ies in 2022.
- Lets record our observations regarding average win_age across all Catgories:
  - The average age of laureates has Increased overtime in the categories of Physics, Chemistry and Medicine.
  - The average age has remained Almost Constant in the categories of Literature and Economics.
  - The average age has Declined overtime in the category of Peace.
- Observe that overall Mean age is Highest for Economics and Lowest for Physics.
- From the boxplot, it looks like the Economic Sciences, Literature and Peace laureates are in general older than those working in Physics, Chemistry and Medicine.

Lets have some Trivia Questions from our dataset.

In [346]:

```
#  Who is the youngest Individual to win the Nobel Prize?
mask = (df_ind["Win_Age"] == df_ind["Win_Age"].min())
df_ind[mask].loc[:, ["Year", "fullname", "Win_Age", "Category"]]
```

Out[346]:

	Year	fullname	Win_Age	Category
877	2014	Malala Yousafzai	17.0	peace

In [348]:

```
#  Who is the oldest Individual to win the Nobel Prize?
mask = (df_ind["Win_Age"] == df_ind["Win_Age"].max())
df_ind[mask].loc[:, ["Year", "fullname", "Win_Age", "Category"]]
```

Out[348]:

	Year	fullname	Win_Age	Category
932	2018	Arthur Ashkin	96.0	physics

In [352]:

```
#  Which Individuals have won the Nobel Prize more than once ?
df_ind["fullname"].value_counts()[df_ind["fullname"].value_counts() > 1]
```

Out[352]:

```
Marie Curie         2
Linus Pauling       2
John Bardeen        2
Frederick Sanger    2
Barry Sharpless     2
Name: fullname, dtype: int64
```

In [354]:

```
#  Which Organizations have won the Nobel Prize more than once ?
df_org["fullname"].value_counts()[df_org["fullname"].value_counts() > 1]

```

Out[354]:

```
International Committee of the Red Cross                        3
Office of the United Nations High Commissioner for Refugees     2
Name: fullname, dtype: int64
```

In [356]:

```
#  Which organization won the Nobel Prize at its youngest age?
df_org[df_org["Win_Age"] == df_org["Win_Age"].min()].loc[:, ["Year", "Category", "fullname", "Laureate_type", "Birth_Year", "Win_Age"
```

Out[356]:

	Year	Category	fullname	Laureate_type	Birth_Year	Win_Age
200	1938	peace	Nansen International Office for Refugees	Organisation	1921	17.0

In [ ]:

```
1
```