Homework 08

⚠️ Before you start ⚠️

Duplicate this Jupyter Notebook in your `week-10` folder (right-click -> Duplicate) and then add your last name to the beginning of it (ie. `blevins-hw-08.ipynb` - otherwise you risk having all your work overwritten when you try to sync your GitHub repository with your instructor's repository.

Overview

In this assignment, you'll synthesize some of the Python skills you've learned over the past month or so, including Pandas and Plotly. You'll be analyzing the opening of new businesses in Colorado during the 1940s.

Draw on the following tutorials:

- 💻 Walsh, Pandas Basics Part 1
- 💻 Walsh, Pandas Basics Part 2
- 💻 Walsh, Pandas Basics Part 3
- 🐼 Pandas Concepts
- 💻 Introduction to Plotly
- 💻 Cleaning Excel Files

The Data

First, get the necessary data files from our shared course repository:

- Open GitHub Desktop and select your course repository (`lastname-sp25-data-materials`)
- Click `Fetch origin` to check for updates
- Go to `Branch` → `Merge into current branch` → select `upstream/main` -> `Merge`
- Click `Push origin` to sync everything up
- Launch Jupyter Lab and navigate to the `week-10` folder

You should see a single Excel file that you will be working with: `co-new-businesses-1940s.xlsx` . Inside that Excel file, there are two separate sheets: `New CO Businesses` and `Cities 1940` .

- `New CO Businesses` : This is a subset of new businesses that were established in Colorado during the 1940s - a subset of data drawn from this database.
- `Cities 1940` : this contains population statistics for Colorado cities in the 1940 Census.

Import Libraries and Load Data

- Import the necessary libraries:
  - pandas (using the alias `pd`)
  - plotly.express (using the alias `px`)

In [7]:
```
import pandas as pd
import plotly.express as px
```

- Load both sheets from the Excel file:
  - Create a variable called `businesses_df` to store the "New CO Businesses" sheet in the Excel file
  - Create a variable called `cities_df` to store the "Cities 1940" sheet in the Excel file
  - Use `pd.read_excel()` with the appropriate parameters

In [9]:
```
businesses_df= pd.read_excel("co-new-businesses-1940s.xlsx", sheet_name= "Ne
cities_df = pd.read_excel("co-new-businesses-1940s.xlsx", sheet_name= 'Citie
```

Familiarize Yourself with the Data

Familiarize yourself with the data:

- Display a sample of 10 rows from each dataframe.
- Check the data types for the columns in each dataframe

In [11]:
```
print(f"{businesses_df.sample(10)}\n")


print(f"{cities_df.sample(10)}\n")

print(f"{cities_df.dtypes}\n")

print(f"{businesses_df.dtypes}\n")
```

```
        entityid                                  Business entity name  \
722  19871116551     MIDDLE PARK WATER USERS PROTECTIVE ASSOCIATION
952  19871110200  CHURCH OF THE ASCENSION & HOLY TRINITY & ASCEN...
4    19871117433     LYNCH-COTTEN POST NO. 190, THE AMERICAN LEGION
834  19871313658  PIONEER ASTRO INDUSTRIES, INC., Dissolved July...
334  19491116218             Morgan Game and Fish Conservation Club
767  19871141764                           RED ROCKS BAPTIST CHURCH
354  19871007393  MELVILLE REALTY COMPANY, INC., Colorado Author...
624  19871113775  UNITED LUMBER AND HARDWARE, INC., Delinquent J...
484  19871109816        THE SOUTH PARK POST 172-THE AMERICAN LEGION
578  19871112651               KIRK OF BONNIE BRAE (CONGREGATIONAL)

                Address              city state  zip_code Country  \
722                 NaN               NaN   NaN       NaN     NaN
952        420 W 18th St           Pueblo    CO   81003.0      US
4         425 Highway 92         Crawford    CO   81415.0      US
834   3410 N PROSPECT ST  COLORADO SPRINGS    CO   80907.0      US
334   427 S. Lake Street       Fort Morgan    CO   80701.0      US
767  14711 W Morrison Rd         Morrison    CO   80465.0      US
354             1 CVS DR       WOONSOCKET    RI    2895.0      US
624       307 E BRIDGE ST         BRIGHTON    CO   80601.0      US
484          602 Clark St         FAIRPLAY    CO   80440.0      US
578      1201 S Steele St           Denver    CO   80210.0      US

    date_entity_formed  year_entity_formed
722         1949-06-06                1949
952         1946-01-30                1946
4           1949-12-30                1949
834         1945-01-08                1945
334         1949-04-11                1949
767         1945-01-11                1945
354         1945-07-12                1945
624         1947-12-30                1947
484         1945-10-24                1945
578         1947-05-08                1947

                city  year  total population
47             delta  1940              3717
185         sedgwick  1940               373
202          timnath  1940               147
182         saguache  1940              1219
119            lamar  1940              4445
88            grover  1940               137
106        jamestown  1940               190
132   manitou springs  1940              1462
81   glenwood springs  1940              2253
73            fowler  1940               922

city                object
year                 int64
total population     int64
dtype: object

entityid                  int64
Business entity name     object
Address                  object
```

```
city                      object
state                     object
zip_code                 float64
Country                   object
date_entity_formed        object
year_entity_formed         int64
dtype: object
```

Data Cleaning and Preparation

Cleaning column names

For both datasets, you want to clean and standardize the column names (headers):

- Change column names to all lowercase
- Replace any whitespace with an underscore (_) - ex. `some column` becomes `some_column`
- *Hint: Use `str.lower()` and `str.replace()`*
- Show the first 10 rows of your dataframe to make sure it worked

In [13]:
```
#Your code here
businesses_df.columns = [col.strip().lower().replace('    ', '_').replace(' '
cities_df.columns = [col.strip().lower().replace('    ', '_').replace(' ', '_


businesses_df.head(10)
cities_df.head(10)
```

Out[13]:

	city	year	total_population
0	akron	1940	1417
1	alamosa	1940	5613
2	alma	1940	469
3	antonito	1940	1220
4	arriba	1940	286
5	arvada	1940	1482
6	aspen	1940	777
7	aurora	1940	3437
8	basalt	1940	212
9	bayfield	1940	372

Standardize and clean data for cities

- Standardize city names in the business data so that it **removes any trailing or**

leading whitespace and **changes the values to all lowercase** (hint: use `.str.strip()` and `.str.lower()`)

- Show the first 10 rows of your dataframe to make sure it worked

In [15]:
```
#Your code here


businesses_df["city"]= businesses_df['city'].str.strip().str.lower()

businesses_df.head(10)
```

Out[15]:

	entityid	business_entity_name	address	city	state	zip_code	countr
0	19871004753	ALAMOSA CREDIT UNION	2437 MAIN ST	alamosa	CO	81101.0	US
1	19871241137	THE UNITED METHODIST CHURCH OF STEAMBOAT SPRINGS	736 OAK ST	steamboat springs	CO	80487.0	US
2	19871275274	ALLIED JEWISH FEDERATION OF COLORADO	300 S. Dahlia St.	denver	CO	80246.0	US
3	19871127721	Iglesia CRISTO REY + Christ the King, ELCA	2300 S Patton Ct	denver	CO	80219.0	US
4	19871117433	LYNCH-COTTEN POST NO. 190, THE AMERICAN LEGION	425 Highway 92	crawford	CO	81415.0	US
5	19871105155	THE BEAR RIVER VALLEY FARMERS COOPERATIVE	193 E Jefferson Ave	hayden	CO	81639.0	US
6	19871162072	Belmar Baptist Church	460 S Kipling St	lakewood	CO	80226.0	US
7	19871110810	Bethel Lutheran Church of Windsor, Colorado	328 Walnut St	windsor	CO	80550.0	US
8	19871116977	BLACKINTON AND DECKER, INC., Delinquent Novemb...	424 LIPAN	denver	CO	80204.0	US
9	19871113871	BOW-MAR OWNERS, INC.	5380 Lakeshore Dr	littleton	CO	80123.0	US

Categorize Cities

Define your function

Create a function called `categorize_city_size` that does the following:

- Takes in a number that corresponds to the population for a city and returns the following based on the size of the city:
  - `Small Town` if population is less than 1,000
  - `Medium Town` if population is between 1,000 to 5,000
  - `Large Town` if population is between 5,000 to 20,000
  - `City` if population greater than or equal to 20,000

In [17]:
```python
def categorize_city_size(population):
    if population>=20000:
        return "City"
    elif population >5000 and population<20000:
        return "Large Town"
    elif population >1000 and population<5000:
        return 'Medium Town'
    else:
        return 'Small Town'
```

Test Your Function

Test out the function on a single number (`2,000`) to make sure it returns `Medium Town`

In [19]:
```python
categorize_city_size(2000)

cities_df.head(10)
```

Out[19]:

	city	year	total_population
0	akron	1940	1417
1	alamosa	1940	5613
2	alma	1940	469
3	antonito	1940	1220
4	arriba	1940	286
5	arvada	1940	1482
6	aspen	1940	777
7	aurora	1940	3437
8	basalt	1940	212
9	bayfield	1940	372

Apply the function

- Take your `cities_df` dataframe and add a new column called `city_category` that applies your function to the `total_population` column of the dataframe.
- *Hint: use* `apply()`
- Show the first 10 rows of your dataframe to make sure it worked

In [21]:
```
cities_df['city_category']= cities_df['total_population'].apply(categorize_c
cities_df.head(10)

businesses_df.head(10)
```

Out[21]:

	entityid	business_entity_name	address	city	state	zip_code	countr
0	19871004753	ALAMOSA CREDIT UNION	2437 MAIN ST	alamosa	CO	81101.0	US
1	19871241137	THE UNITED METHODIST CHURCH OF STEAMBOAT SPRINGS	736 OAK ST	steamboat springs	CO	80487.0	US
2	19871275274	ALLIED JEWISH FEDERATION OF COLORADO	300 S. Dahlia St.	denver	CO	80246.0	US
3	19871127721	Iglesia CRISTO REY + Christ the King, ELCA	2300 S Patton Ct	denver	CO	80219.0	US
4	19871117433	LYNCH-COTTEN POST NO. 190, THE AMERICAN LEGION	425 Highway 92	crawford	CO	81415.0	US
5	19871105155	THE BEAR RIVER VALLEY FARMERS COOPERATIVE	193 E Jefferson Ave	hayden	CO	81639.0	US
6	19871162072	Belmar Baptist Church	460 S Kipling St	lakewood	CO	80226.0	US
7	19871110810	Bethel Lutheran Church of Windsor, Colorado	328 Walnut St	windsor	CO	80550.0	US
8	19871116977	BLACKINTON AND DECKER, INC., Delinquent Novemb...	424 LIPAN	denver	CO	80204.0	US
9	19871113871	BOW-MAR OWNERS, INC.	5380 Lakeshore Dr	littleton	CO	80123.0	US

Analyze Businesses by Year

Let's take a look at how many new businesses were formed in Colorado in each year during the 1940s:

Calculate new businesses by year

Create a variable called `businesses_per_year` by:

- Counting the number of new businesses based on `year_entity_formed`
- *Hint: use `value_counts()` and `reset_index()`*
- Show the first 10 rows of your dataframe

In [23]:
```
businesses_per_year= businesses_df[['year_entity_formed']].value_counts().re

businesses_per_year.head(10)
```

Out[23]:

	year_entity_formed	count
0	1947	161
1	1948	156
2	1946	153
3	1949	133
4	1945	87
5	1940	72
6	1941	69
7	1943	47
8	1944	43
9	1942	35

Visualize new businesses by year

Create a bar chart using Plotly Express showing new businesses per year:

- Set x-axis to the year
- Set y-axis to the number of new businesses
- Add an appropriate title and labels
- Display text on each bar
- Hint: Use `px.bar()`

In [25]:
```
fig = px.bar(
    businesses_per_year,
    x='year_entity_formed',
    y='count',
    title='Number of Businesses Formed in Colorado Per Year',  # Add a title
    labels={'year_entity_formed': 'year', 'count': '# of Busnesses'}, # Rena
    color='count',
    color_continuous_scale='Agsunset_r',
    template='xgridoff'
)

fig.show()
```

Analyze Businesses by City

Let's take a look at how many new businesses were formed in each Colorado city during the 1940s:

Calculate number of new businesses by city

Create a new variable called `city_businesses` that contains:

- A dataframe with counts of the number of new businesses in each city
- *Hint: Use `value_counts()` and `reset_index()`*
- Show the first 10 rows of your dataframe

In [27]:
```
#Your code here

city_businesses= businesses_df["city"].value_counts().reset_index()
city_businesses.head(10)
```

Out[27]:

	city	count
0	denver	152
1	colorado springs	34
2	lakewood	22
3	pueblo	20
4	arvada	14
5	grand junction	14
6	fort collins	13
7	greeley	13
8	centennial	12
9	englewood	12

Visualize new businesses by city

Create a bar chart with Plotly Express showing the top 10 cities with the most new businesses created during the 1940s:

- Filter to only show the top 10 cities (hint: use `.head()`)
- Set x-axis to `city`
- Set y-axis to `count`
- Add an appropriate title and labels

In [29]:
```
fig = px.bar(
    city_businesses.head(10),
    x='city',
    y='count',
    title='Top 10 Cities With the Most New Businesses during the 1940s',  #
    labels={'city': 'City', 'count': '# of Busnesses'}, # Rename axis labels
    color='city',
    color_continuous_scale='Agsunset_r',
    template='xgridoff'

)

fig.show()
```

Combine Business and City Data

We have two datasets, both of which contain information about Colorado cities. Let's combine the two into a single dataframe that contains both information about new businesses and their population in the 1940 census.

Merge dataframes

Merge the two dataframes together:

- Create a new variable called `merged_df`
- Use `pd.merge()` on the `city_businesses and cities_df` dataframes
- Figure out which column is shared between the two to use as your "key" to merge them
- ⚠️ **Note: use the `how='inner'` parameter for your merge**
- Show the first 10 rows of your new dataframe

```
In [31]: #Your code here
```

```
merged_df = pd.merge(
    city_businesses,
    cities_df,
    on='city',
    how='inner'
)

merged_df.head(10)
```

Out[31]:

	city	count	year	total_population	city_category
0	denver	152	1940	322412	City
1	colorado springs	34	1940	36789	City
2	pueblo	20	1940	52162	City
3	arvada	14	1940	1482	Medium Town
4	grand junction	14	1940	12479	Large Town
5	fort collins	13	1940	12251	Large Town
6	greeley	13	1940	15995	Large Town
7	englewood	12	1940	9680	Large Town
8	littleton	11	1940	2244	Medium Town
9	aurora	10	1940	3437	Medium Town

Filter out missing values

You'll note that several rows of data contain `NaN` or missing values - this means that there was a city listed in the businesses dataframe but it didn't have a corresponding match in the population dataframe. For now, remove these from the `merged_df` dataframe:

- Filter out rows where `total_population` is NaN
- *Hint: use a filter + `.notna()`*

In [33]:
```
merged_df= merged_df[merged_df['total_population'].notna()]
```

Calculate new businesses on a per capita rate

To make it easier to compare larger cities with smaller cities, you're going to calculate a new column for each city: the number of new businesses per 1,000 residents.

- Add a new column to `merged_df` called `biz_per_thousand` that is filled with:
  - A calculation dividing the `count` column by the `total_population` column and multiplying by 1,000
- Sort the merged dataframe by `biz_per_thousand` in descending order

- Show the first 10 rows of the dataframe to check if it worked

In [35]:
```
merged_df['biz_per_thousand']= merged_df['count']/merged_df['total_populatio

merged_df = merged_df.sort_values(by='biz_per_thousand', ascending=False)

merged_df.head(10)
```

Out[35]:

	city	count	year	total_population	city_category	biz_per_thousand
92	green mountain falls	1	1940	87	Small Town	11.494253
36	keenesburg	3	1940	284	Small Town	10.563380
59	bennett	2	1940	199	Small Town	10.050251
3	arvada	14	1940	1482	Medium Town	9.446694
52	morrison	2	1940	216	Small Town	9.259259
20	castle rock	5	1940	580	Small Town	8.620690
33	woodland park	3	1940	372	Small Town	8.064516
54	granby	2	1940	251	Small Town	7.968127
79	grover	1	1940	137	Small Town	7.299270
96	timnath	1	1940	147	Small Town	6.802721

Visualize new business creation by city

Let's say we want to see the cities with the highest *rate* of business creation (ie. new businesses per thousand residents)

- Create a bar chart in Plotly of `merged_df` :
  - Filter to only show the top 10 cities (use `.head(10)`)
  - Set x-axis to `city`
  - Set y-axis to `biz_per_thousand`
  - Use `city_category` for color
  - Add an appropriate title and labels

In [37]:
```
fig = px.bar(
    merged_df.head(10),
    x='city',
    y='biz_per_thousand',
    title='Cities with highest new businesses per thousand residents',  # Ad
    labels={'city': 'City', 'biz_per_thousand': '# of Busnesses per 1000 res
    color='biz_per_thousand',
    color_continuous_scale='Inferno',
    template='xgridoff'
)

fig.show()
```

Bonus: New businesses by city category

Let's say we want to compare different size categories to see whether new businesses were cropping up in smaller places or bigger cities.

Create a new dataframe

First, you'll need to create a new dataframe that consists of four rows, with each row a different category of city containing the total number of businesses created within that category of city.

- Create a new dataframe called `city_category_totals`
- Start with `merged_df`
- Group by `city_category`
- Add up (`sum()`) the `count` column

- Use `.reset_index()`

In [39]:
```
#Your code here
```

Visualize businesses by city category

- Create a pie chart in Plotly:
  - Use `px.pie()` with appropriate parameters
  - Use `city_category_totals` as your dataframe
  - Use `count` for your values
  - Use `city_category` for your names
  - Add an appropriate title and labels

In [41]:
```
#Your code here
```

Bonus Challenge: Create a Scatterplot

Create a scatter plot in Plotly showing:

- The relationship between city population (x-axis) and new businesses (y-axis)
- Only data for towns with a population of 2,000 or more people.
- Dots sized according to the number of new businesses in that city
- Dots colored according to their size category

In [43]:
```
#Your code here
```

Submission Guidelines

- Run all code cells and make sure it is outputting without errors
- Submit both the notebook file (.ipynb) and a PDF export of your notebook on Canvas
- Note: the PDF probably won't display the Plotly figures - that's okay