Understanding Socio-Economic Dynamics in France:

INSIGHTS INTO POPULATION, ESTABLISHMENTS, SALARIES, AND GENDER INEQUALITIES

Stéphanie Tuyêt-Hông Pragassam
IRONHACK | DATA ANALYTICS | APRIL 2024

Table of content

Introduction

In today's data-driven world advanced analytics, exploring large datasets offers unparalleled opportunities to understand societal complexities. This project dives into the intricate socio-economic fabric of France, leveraging datasets provided by the French National Institute of Statistics and Economic Studies (INSEE). Through an analysis of population demographics, establishment distributions, salary structures, and job categories across French departments and regions, we seek to uncover patterns, trends, and correlations that offer valuable insights into the socio-economic landscape of France.

With access to INSEE datasets, our project aims to address fundamental questions about resource allocation, employment trends, and gender dynamics in the French labor market. By employing rigorous data analysis, our goal is to provide valuable insights to inform decision-making and drive socio-economic progress.

As we embark on this journey, we cannot ignore the issue of gender wage disparities and the underrepresentation of women in managerial positions in France. By examining salary data and analyzing the presence of women in leadership roles, we aim to shed light on persistent inequalities in the labor market. Through our analysis, we hope to contribute to discussions on gender equality and advocate for measures to promote women's advancement in the workplace.

Furthermore, our project delves into job categories to understand employment patterns and socio-economic stratification in France. By examining the distribution of individuals across different job categories and exploring the representation of women in managerial positions within these categories, we seek to uncover patterns of occupational mobility and socio-economic disparities. Our analysis aims to provide insights into the composition of the French workforce and the challenges faced by women in accessing leadership roles. From demographic shifts and employment patterns to income distributions and regional disparities, our analysis endeavors to paint a comprehensive picture of the socio-economic dynamics shaping French society.

Embark on a journey through the intricate web of data, revealing hidden insights and advancing towards a more equitable and inclusive future for all in France.

The goals of this project are to:

- Explore demographic trends and disparities in France
- Analyze salary distributions across different demographics and job categories
- Investigate the representation of women in managerial positions
- Identify correlations between population demographics and establishment data

This project is available on this repository : https://github.com/steprag/fp_finale_project

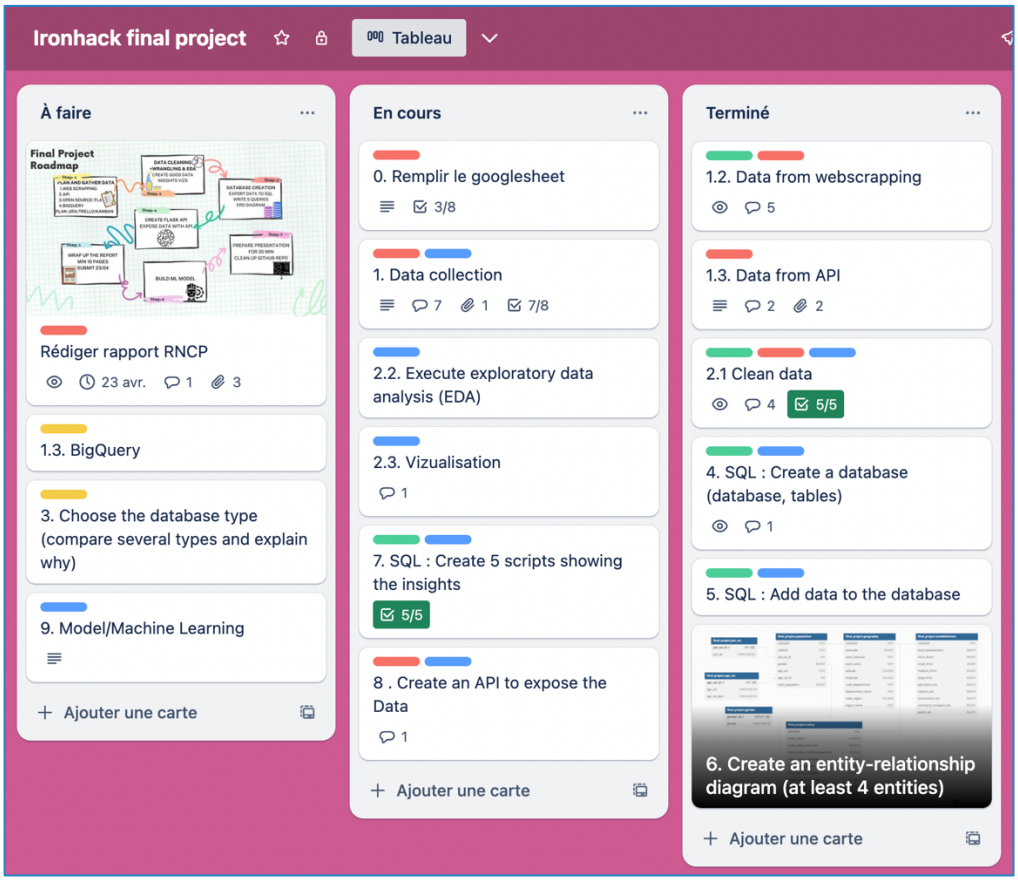
Project Management :

High level plan :

- Research about project topic
- Data collection
- Project scope
- Project planning in Trello
- Exploratory data analysis in Python
    - data wrangling,
    - data cleaning
    - data visualization (Python + Tableau)
- Selection and creation of a database using MySQL
- Adding data to database and create Entity Relationship Diagram
- Data manipulation in SQL
- Exposing data via API
- Machine Learning : train and test models – for presentation

Project Planning :

Trello board to manage daily project tasks:



I. Data sources & Data collection

1. Data from flat files

We sourced our dataset from reliable sources, including the National Institute of Statistics and Economic Studies (INSEE) and the data.gouv.fr platform.

Here's a summary of our data collection:

1. **Number of active establishments as of the end of 2020 grouped into 5 major sectors** :
    - This dataset provides information on the number of active establishments categorized into five major sectors. It offers insights into the distribution of establishments across different sectors at the end of the year 2020.
    - https://www.insee.fr/fr/statistiques/4991205
    - 'etablissement.csv'

2. **Average net hourly wage by socio-professional category, gender, and age in 2020** :
    - This dataset offers details on the average net hourly wage based on socio-professional category, gender, and age in the year 2020. It enables analysis of wage disparities across various socio-demographic groups.
    - https://www.insee.fr/fr/statistiques/2021266
    - 'salaire_par_commune.csv'

3. **Population aged 15 years and older by gender, age, and socio-professional category** :
    - This dataset presents demographic information on the population aged 15 years and older, segmented by gender, age, and socio-professional category. It allows for a comprehensive examination of the demographic composition and socio-economic characteristics of the population.
    - https://www.insee.fr/fr/statistiques/7631680?sommaire=7632456
    - 'population.csv'

4. **Communes de France - Base of postal codes** :
    - This dataset, provided by La Poste, includes the names of regions, departments, and municipalities in lowercase letters, along with their respective INSEE codes. It serves as a valuable resource for geographic analysis and mapping.
    - https://www.data.gouv.fr/fr/datasets/communes-de-france-base-des-codes-postaux/
    - 'communes-departement-region.csv'

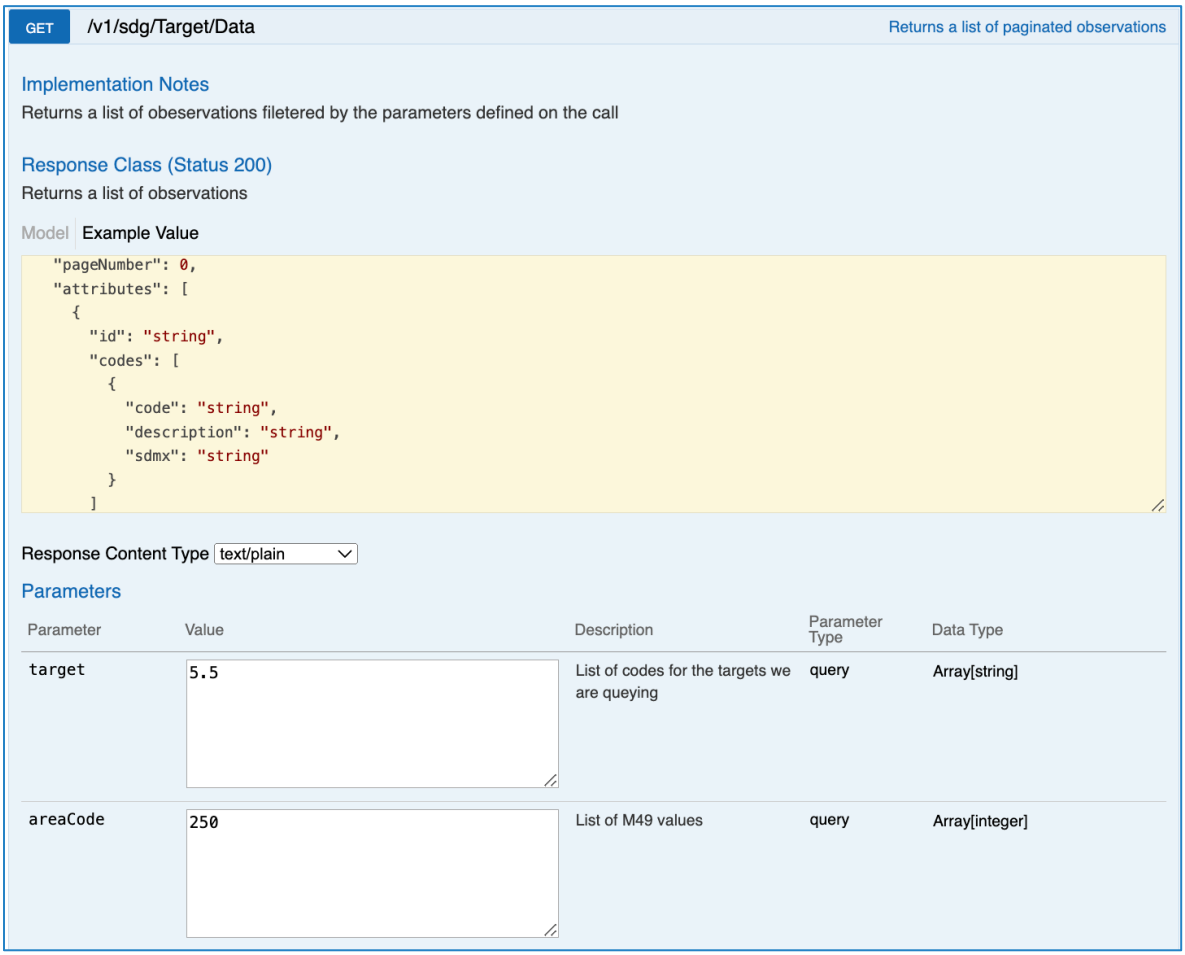
We specifically chose data from the year 2020 to maintain consistency throughout our dataset, as certain data for subsequent years was not available. This decision ensures coherence, reliability and completeness in our analysis and allows for meaningful comparisons across different variables and indicators.

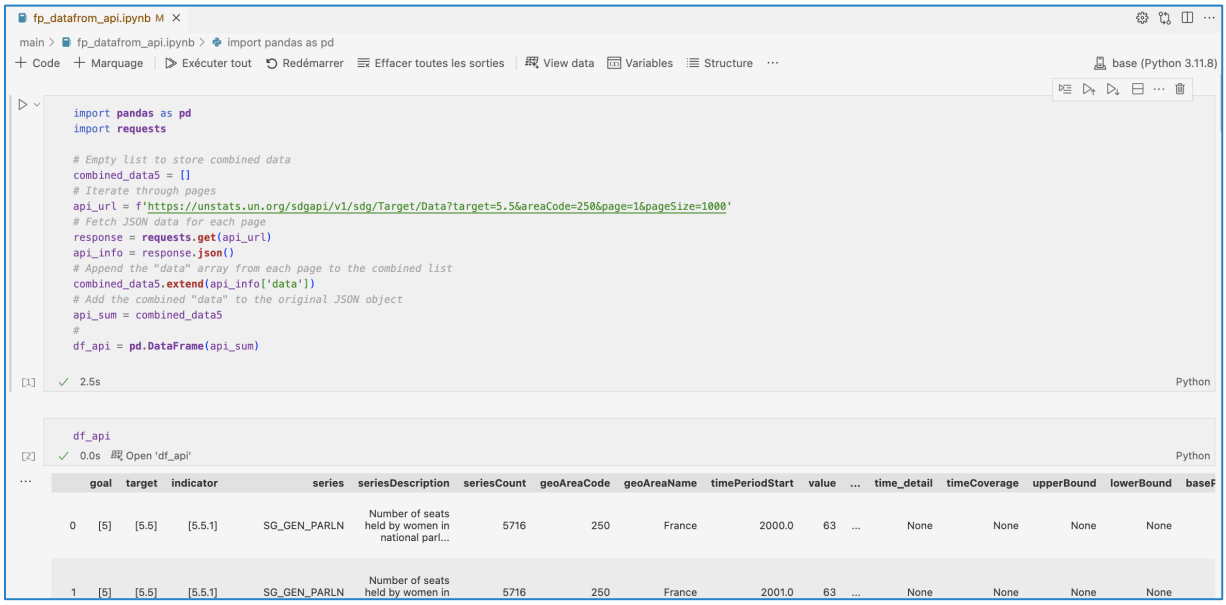
2. Data from API

The data collection process also involved extracting relevant information from the United Nations Statistics Division SDG API, specifically focusing on target ‘5.5’, which pertains to ensuring women's full and effective participation and equal opportunities for leadership at all levels of decision-making, and on the area code ‘250’ for France.

This API enables an access to data related to the proportion of women holding managerial positions in France. By leveraging this API, we can retrieve valuable insights into the representation of women in leadership roles. This dataset from the SDG API complements our other sources of data and contributes to a more comprehensive analysis of gender equality and women's empowerment in France.

- SDG API link : https://unstats.un.org/sdgapi/swagger/#!/Target/V1SdgTargetDataGet
- Target for the API : https://sdgs.un.org/goals/goal5#targets_and_indicators
- Data from API : ‘api_data5.5_raw.csv’



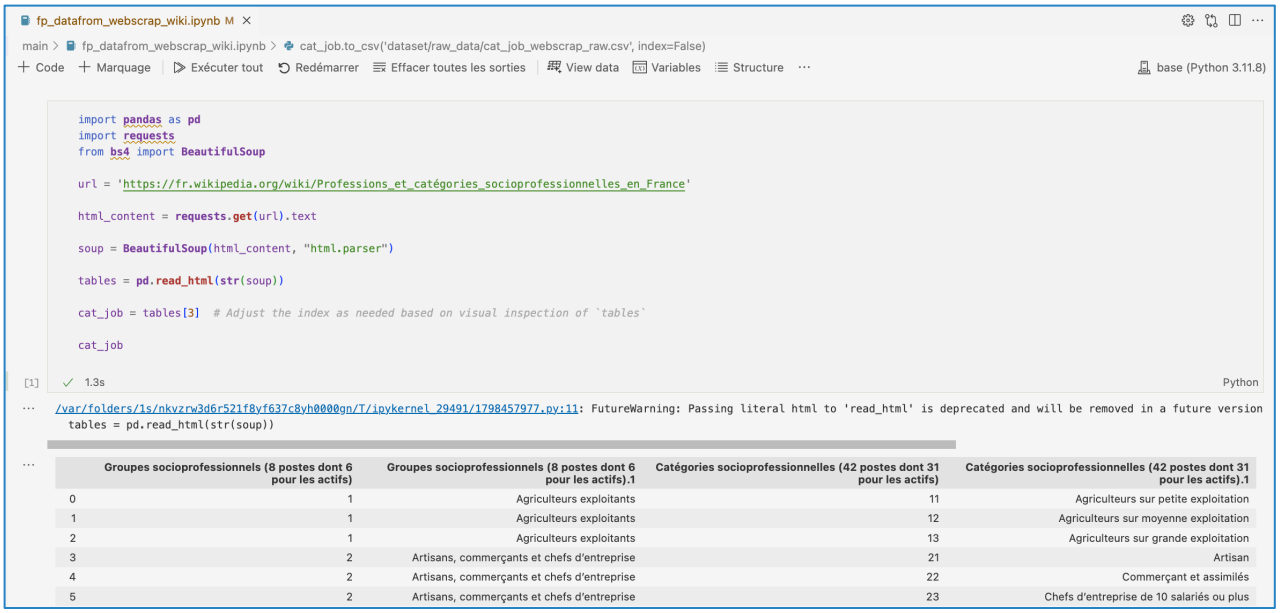


3. Data from web scraping

For the collection of data on professions and socio-professional categories (PCS), we utilized web scraping techniques to extract information from the Wikipedia page titled "Professions et catégories socioprofessionnelles en France." Specifically, we focused on retrieving data from the third table on the page, which provides the correspondence between socio-professional categories and socio-professional groups.

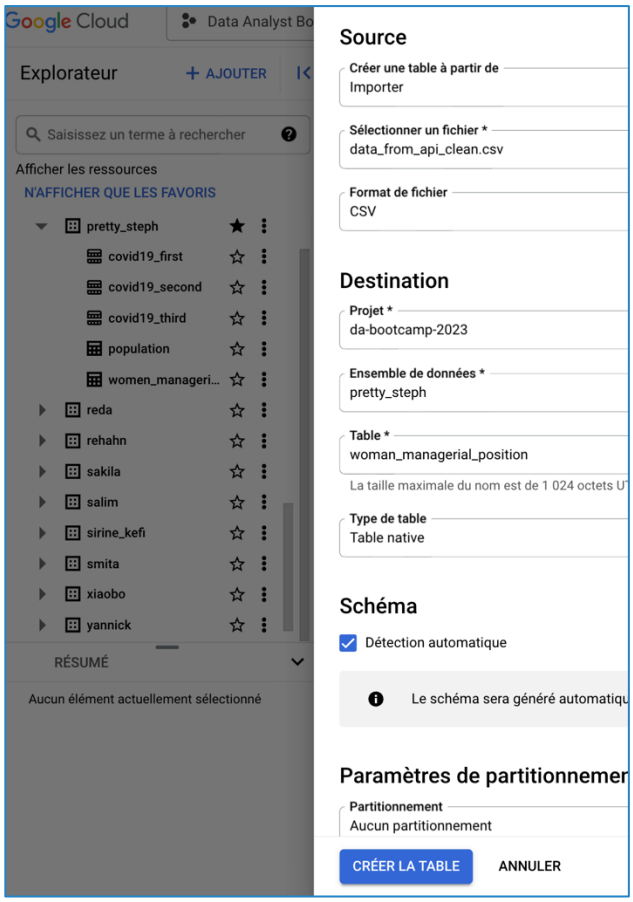
The professions and socio-professional categories (PCS) serve as a statistical classification system used to categorize occupations. This classification was established by the National Institute of Statistics and Economic Studies (INSEE) in 1982. By scraping this data from Wikipedia, we were able to obtain more insights into the classification of professions and socio-professional categories, which are essential for our analysis of various socio-economic factors in France.

- URL: https://fr.wikipedia.org/wiki/Professions_et_cat%C3%A9gories_socioprofessionnelles_en_France
- Data from webscraping : 'cat_job_webscrap_raw.csv'

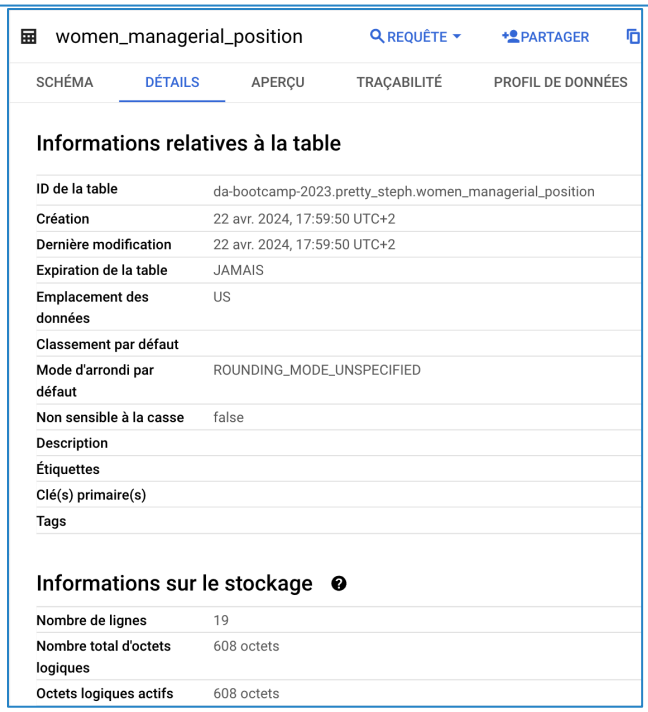


4. Data from Big Query

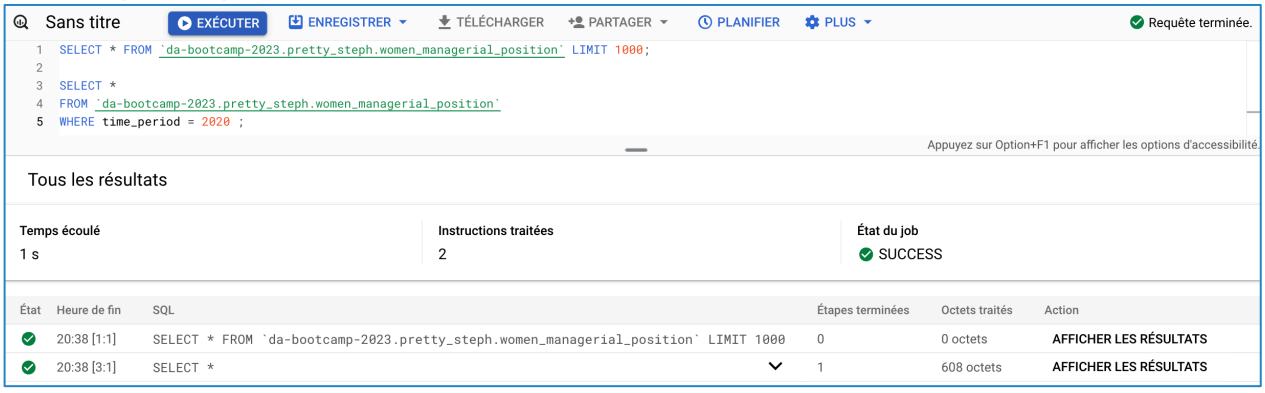
I was unable to locate an appropriate database on BigQuery, I opted to upload a cleaned CSV file to BigQuery for query execution. I uploaded it directly through the console and created a new table :

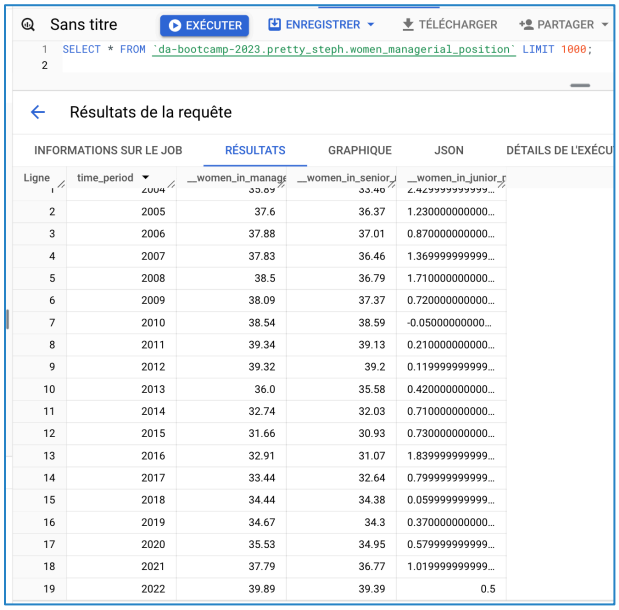


- Information on the schema setup and detailed description of the data structure :



- Examples of queries:





Ligne	time_period	__women_in_manage	__women_in_senior_	__women_in_junior_r
1	2004	33.69	33.46	2.429999999999...
2	2005	37.6	36.37	1.230000000000...
3	2006	37.88	37.01	0.870000000000...
4	2007	37.83	36.46	1.369999999999...
5	2008	38.5	36.79	1.710000000000...
6	2009	38.09	37.37	0.720000000000...
7	2010	38.54	38.59	-0.05000000000...
8	2011	39.34	39.13	0.210000000000...
9	2012	39.32	39.2	0.119999999999...
10	2013	36.0	35.58	0.420000000000...
11	2014	32.74	32.03	0.710000000000...
12	2015	31.66	30.93	0.730000000000...
13	2016	32.91	31.07	1.839999999999...
14	2017	33.44	32.64	0.799999999999...
15	2018	34.44	34.38	0.059999999999...
16	2019	34.67	34.3	0.370000000000...
17	2020	35.53	34.95	0.579999999999...
18	2021	37.79	36.77	1.019999999999...
19	2022	39.89	39.39	0.5

```
3  SELECT *
4  FROM `da-bootcamp-2023.pretty_steph.women_managerial_position`
5  WHERE time_period = 2020 ;
```

Résultats de la requête

INFORMATIONS SUR LE JOB | RÉSULTATS | GRAPHIQUE | JSON | DÉTAILS DE L'EXÉCUTION

Ligne	time_period	__women_in_manage	__women_in_senior_	__women_in_junior_r
1	2020	35.53	34.95	0.579999999999...

II. Data cleaning & Exploratory data analysis

1. Data cleaning and wrangling

Once the data collection phase was completed, the subsequent task involved cleaning the gathered data to construct the database and ensure its readiness for utilization in both the API and the machine learning model.

Several data cleaning procedures were executed, including ;

- Checking and handling the data types :

```
salary.dtypes
```

```
CODGEO        object
SNHM20       float64
SNHMC20      float64
SNHMP20      float64
SNHME20      float64
```

```
df_api_pivot['%_women_in_managerial_positions'] = df_api_pivot['%_women_in_managerial_positions'].astype(float)
df_api_pivot['%_women_in_senior_middle_management'] = df_api_pivot['%_women_in_senior_middle_management'].astype(float)
```

- Handling null values

```
print(population.isnull().sum())
print(salary.isnull().sum())
print(establishment.isnull().sum())
print(geography.isnull().sum())
```

✓ 0.0s

```
CODGEO        0
LIBGEO        0
job_cat       0
gender        0
```

```
salary.isna().sum()
```

✓ 0.0s

```
CODGEO        0
SNHM20        0
SNHMC20       0
SNHMP20       0
SNHME20       0
```

```
df_api_pivot.dropna(inplace=True)
```

```
geography[geography['CODGEO'] == '27676']
```

✓ 0.0s

	CODGEO	postcode	town_fullname	town_name	latitude	longitude	code_departement	departement_name	code_region	region_name
11098	27676	27940	Les Trois Lacs	Trois Lacs	49.20122	1.295948	27	Eure	28.0	Normandie

```
# Remplacer les valeurs NA de LIBGEO par 'Les Trois Lacs' pour CODGEO égal à '27676'
population_csp.loc[population_csp["CODGEO"] == '27676', "LIBGEO"] = population_csp.loc[population_csp["CODGEO"] == '27676', "LIBGEO"].fillna('Les Trois Lacs')

# Afficher le DataFrame mis à jour
population_csp[population_csp['CODGEO'] == '27676']
```

- Checking and handling duplicates

```
geography.duplicated().any()
```

✓ 0.0s

```
True
```

```
geography_clean = geography.drop_duplicates()
```

- Checking unique values :

```
print(population.nunique())
print(geography.nunique())
print(salary.nunique())
print(establishment.nunique())
```

✓ 0.1s

- Cleaning columns names, dropping & reorganizing columns :

```
salary = salary[['CODGEO', 'SNHM20', 'SNHMH20', 'SNHMF20','SNHMC20','SNHMFC20','SNHMHC20','SNHMP20',

salary = salary.rename(columns = {
                    "SNHM20": "mean_salary",
                    "SNHMH20": "mean_salary_male",
                    "SNHMF20": "mean_salary_female",
                    "SNHMC20": "mean_salary_executive",
                    "SNHMFC20":"mean_salary_executive_female",
                    "SNHMHC20":"mean_salary_executive_male",
                    "SNHMP20":"mean_salary_middlemanagement",
                    "SNHMFP20":"mean_salary_middlemanagement_female",
                    "SNHMHP20":"mean_salary_middlemanagement_male",
                    "SNHME20":"mean_salary_employee",
                    "SNHMFE20":"mean_salary_employee_female",
                    "SNHMHE20":"mean_salary_employee_male",
                    "SNHMO20":"mean_salary_worker",
                    "SNHMFO20":"mean_salary_worker_female",
                    "SNHMHO20":"mean_salary_worker_male",
                    "SNHM1820":"mean_salary_youngage",
                    "SNHMF1820":"mean_salary_youngage_female",
                    "SNHMH1820":"mean_salary_youngage_male",
                    "SNHM2620":"mean_salary_mediumage",
                    "SNHMF2620":"mean_salary_mediumage_female",
                    "SNHMH2620":"mean_salary_mediumage_male",
                    "SNHM5020":"mean_salary_oldage",
                    "SNHMF5020":"mean_salary_oldage_female",
                    "SNHMH5020":"mean_salary_oldage_male",

                    })
```

```
cat_job.rename(columns={
    'Groupes socioprofessionnels (8 postes dont 6 pour les actifs)': 'cat_job_id',
    'Groupes socioprofessionnels (8 postes dont 6 pour les actifs).1': 'french_cat_job',
    'Catégories socioprofessionnelles (42 postes dont 31 pour les actifs)': 'sub_cat_job_id',
    'Catégories socioprofessionnelles (42 postes dont 31 pour les actifs).1': 'french_sub_cat_job'
}, inplace=True)
```

```
    geography = geography.rename(columns = {"code_commune_INSEE": "CODGEO",
                        "code_postal": "postcode",
                        "nom_commune_complet": "town_fullname",
                        "nom_commune": "town_name",
                        "nom_departement": "departement_name",
                        "nom_region":"region_name",
                        })
```

```
salary = salary.loc[:, ['CODGEO', 'gender', 'mean_salary', 'mean_salary_executive',
                        'mean_salary_middlemanagement', 'mean_salary_employee',
                        'mean_salary_worker', 'mean_salary_youngage',
                        'mean_salary_mediumage', 'mean_salary_oldage']]
```

```
population_final.drop(columns=['NIVGEO'], inplace=True)
```

- Value formatting

```
df_api_pivot['%_women_in_managerial_positions'] = df_api_pivot['%_women_in_managerial_positions'].round(2)
df_api_pivot['%_women_in_senior_middle_management'] = df_api_pivot['%_women_in_senior_middle_management'].round(2)
```

```
# Arrondir les valeurs de la colonne 'population' à des chiffres entiers
population['total_population'] = population['total_population'].round().astype(int)
```

✓ 0.2s

```
correspondance_cat = {
    '1': 'farmer',
    '2': 'artisan_merchant_entrepreneur',
    '3': 'executive',
    '4': 'middle_management',
    '5': 'employee',
    '6': 'worker',
    '7': 'retired',
    '8': 'unemployed',
}

# Remplacer les valeurs numériques dans la colonne 'job_cat' par les catégories correspondantes
population_csp_clean['job_cat'] = population_csp_clean['job_cat'].replace(correspondance_cat)
```

```
# Sélectionner toutes les colonnes sauf 'CODGEO' et 'gender'
colonnes_a_arrondir = [colonne for colonne in salary.columns if colonne not in ['CODGEO', 'gender']]

# Arrondir les valeurs des colonnes sélectionnées au deuxième décimale
salary[colonnes_a_arrondir] = salary[colonnes_a_arrondir].round(2)

# Remplacer les valeurs 1 par 'male' et 2 par 'female' dans la colonne 'gender'
salary['gender'] = salary['gender'].replace({1: 'male', 2: 'female'})
```

- Reducing the number of categories

By creating larger categories or bins ：

```
# Définition de la fonction pour catégoriser les âges
def categoriser_age(age):
    if age >= 15 and age <= 24:
        return 'young'
    elif age >= 25 and age <= 49:
        return 'medium'
    else:
        return 'old'

# Appliquer la fonction à la colonne 'age' pour créer la nouvelle colonne 'age_cat'
population['age_cat'] = population['age'].apply(categoriser_age)

population.head()
```

0.2s

NIVGEO	CODGEO	LIBGEO	job_cat	age	gender	total_population	age_cat
COM	1001	L'Abergement-Clémenciat	farmer	55	1	5	old
COM	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	20	2	5	young
COM	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	30	1	5	medium
COM	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	40	2	5	medium
COM	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	45	1	5	medium

```
# Grouper par les colonnes spécifiées et sommer les valeurs de 'total_population'
population_clean = population.groupby(['CODGEO', 'LIBGEO', 'job_cat', 'gender', 'age_cat'], as_index=False)['total_population'].sum()

# Afficher les premières lignes du DataFrame résultant
population_clean.head()
```

0.8s Open 'population_clean'

	CODGEO	LIBGEO	job_cat	gender	age_cat	total_population
0	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	1	medium	10
1	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	1	old	14
2	1001	L'Abergement-Clémenciat	artisan_merchant_entrepreneur	2	medium	5

```
establishment['micro_firms'] = establishment['ET_TOT_1_4'] + establishment['ET_TOT_5_9']
establishment['small_firms'] = establishment['ET_TOT_10_19'] +  establishment['ET_TOT_20_49']
establishment['medium_firms'] = establishment['ET_TOT_50_99'] + establishment['ET_TOT_100_199']
establishment['large_firms'] = establishment['ET_TOT_200_499'] + establishment['ET_TOT_500P']

establishment = establishment.rename(columns = {#"CODGEO": "area_code",
                                                "ET_TOT": "total_establishment",
                                                "ET_AZ": "agriculture_est",
                                                "ET_BE": "industry_est",
                                                "ET_FZ": "construction_est",
                                                "ET_GU":"commerce_transport_est",
                                                "ET_OQ":"public_est",

                                                })
```

- Data homogenization and normalization

Here, we want to separate salary by gender in 2 data frames, and then concatenate them backtogether after homogenizing the column names:

```
salary_male = salary[['CODGEO','mean_salary_male','mean_salary_executive_male','mean_salary_middlemanageme
salary_male['gender']= 1

salary_male = salary_male.rename(columns = {
                                "mean_salary_male":"mean_salary",
                                "mean_salary_executive_male":"mean_salary_executive",
                                "mean_salary_middlemanagement_male":"mean_salary_middlemanagement",
                                "mean_salary_employee_male":"mean_salary_employee",
                                "mean_salary_worker_male":"mean_salary_worker",
                                "mean_salary_youngage_male":"mean_salary_youngage",
                                "mean_salary_mediumage_male":"mean_salary_mediumage",
                                "mean_salary_oldage_male":"mean_salary_oldage",

                                })
```

```
salary_female = salary[['CODGEO','mean_salary_female','mean_salary_executive_female','mean_salary_middlemanag
salary_female['gender']= 2

salary_female = salary_female.rename(columns = {
                                "mean_salary_female":"mean_salary",
                                "mean_salary_executive_female":"mean_salary_executive",
                                "mean_salary_middlemanagement_female":"mean_salary_middlemanagement",
                                "mean_salary_employee_female":"mean_salary_employee",
                                "mean_salary_worker_female":"mean_salary_worker",
                                "mean_salary_youngage_female":"mean_salary_youngage",
                                "mean_salary_mediumage_female":"mean_salary_mediumage",
                                "mean_salary_oldage_female":"mean_salary_oldage",

                                })
```

```
salary_concat = pd.concat([salary_female, salary_male]).sort_values(by="CODGEO")
salary_concat
```

✓ 0.0s Open 'salary_concat'

	CODGEO	mean_salary	mean_salary_executive	mean_salary_middlemanagement	mean_salary_em
0	01004	13.043695	21.806580	14.368130	11.0
0	01004	16.202197	26.693345	18.274408	11.
1	01007	16.406488	25.832202	17.070232	11.7

Shape before :

```
salary.shape
```

✓ 0.0s

```
5421, 25)
```

Shape after :

```
salary_concat.shape
```

✓ 0.0s

```
(10842, 10)
```

- Filtering on columns, pivot the table, and or adding columns with calculation to get the data and the data frame wanted :

```
#·Créer·un·DataFrame·avec·les·colonnes·spécifiées
df_api_filtered = df_api_filtered.loc[:, ['seriesDescription', 'timePeriodStart', 'value']].copy()

# Afficher le DataFrame filtré
df_api_filtered
```

	seriesDescription	timePeriodStart	value
98	Proportion of women in managerial positions - ...	2000.0	34.97
99	Proportion of women in managerial positions - ...	2001.0	35.51
100	Proportion of women in managerial positions - ...	2002.0	36.43
101	Proportion of women in managerial positions - ...	2003.0	35.41
102	Proportion of women in managerial positions - ...	2004.0	35.89

```
df_api_pivot = df_api_filtered.pivot(index='time_period', columns='serie_description', values='value')
df_api_pivot
```

serie_description time_period	Proportion of women in managerial positions - 19th ICLS (%)	Proportion of women in senior and middle management positions - 19th ICLS (%)
2000	34.97	NaN
2001	35.51	NaN
2002	36.43	NaN
2003	35.41	NaN
2004	35.89	33.46

```
# Convertir les colonnes en types numériques
df_api_pivot['%_women_in_managerial_positions'] = pd.to_numeric(df_api_pivot['%_women_in_managerial_positions'], errors='coerce')
df_api_pivot['%_women_in_senior_middle_management'] = pd.to_numeric(df_api_pivot['%_women_in_senior_middle_management'], errors='coerce')

# Calculer la différence entre les deux colonnes et l'assigner à une nouvelle colonne
df_api_pivot['%_women_in_junior_management'] = df_api_pivot['%_women_in_managerial_positions'] - df_api_pivot['%_women_in_senior_middle_management']

df_api_pivot
```

✓ 0.0s Open 'df_api_pivot'

serie_description time_period	%_women_in_managerial_positions	%_women_in_senior_middle_management	%_women_in_junior_management
2004	35.89	33.46	2.43
2005	37.60	36.37	1.23

2. Cleaned datasets summary

At the end of these processes, our clean datasets are :

```
salary = pd.read_csv('dataset/clean_data/salary_concat.csv', sep=',')
salary_all = pd.read_csv('dataset/clean_data/salary_final.csv', sep=',')
population = pd.read_csv('dataset/clean_data/population_clean.csv', sep=',')
population_all = pd.read_csv('dataset/clean_data/population_final.csv', sep=',')
geography = pd.read_csv('dataset/clean_data/geography_clean.csv', sep=',')
establishment = pd.read_csv('dataset/clean_data/establishment_clean.csv', sep=',')
```

✓ 1.0s

```
woman_managerial_position = pd.read_csv('dataset/clean_data/data_from_api_clean.csv', sep=',')
french_cat_job = pd.read_csv('dataset/clean_data/cat_job_webscrap.csv', sep=',')
```

✓ 0.0s

The datasets now consist of the following columns :

```
print(population.columns)
print(salary.columns)
print(establishment.columns)
print(geography.columns)
```

✓ 0.0s

```
Index(['CODGEO', 'LIBGEO', 'job_cat', 'gender', 'age_cat', 'total_population'], dtype='object')
Index(['CODGEO', 'mean_salary', 'mean_salary_executive',
       'mean_salary_middlemanagement', 'mean_salary_employee',
       'mean_salary_worker', 'mean_salary_youngage', 'mean_salary_mediumage',
       'mean_salary_oldage', 'gender'],
      dtype='object')
Index(['CODGEO', 'total_establishment', 'micro_firms', 'small_firms',
       'medium_firms', 'large_firms', 'agriculture_est', 'industry_est',
       'construction_est', 'commerce_transport_est', 'public_est'],
      dtype='object')
Index(['CODGEO', 'postcode', 'town_fullname', 'town_name', 'latitude',
       'longitude', 'code_departement', 'departement_name', 'code_region',
       'region_name'],
      dtype='object')
```

```
print(population_all.columns)
print(salary_all.columns)
```
✓ 0.0s

```
Index(['CODGEO', 'LIBGEO', 'job_cat', 'age', 'gender', 'total_population',
       'age_cat'],
      dtype='object')
Index(['CODGEO', 'mean_salary', 'mean_salary_male', 'mean_salary_female',
       'mean_salary_executive', 'mean_salary_executive_female',
       'mean_salary_executive_male', 'mean_salary_middlemanagement',
       'mean_salary_middlemanagement_female',
       'mean_salary_middlemanagement_male', 'mean_salary_employee',
       'mean_salary_employee_female', 'mean_salary_employee_male',
       'mean_salary_worker', 'mean_salary_worker_female',
       'mean_salary_worker_male', 'mean_salary_youngage',
       'mean_salary_youngage_female', 'mean_salary_youngage_male',
       'mean_salary_mediumage', 'mean_salary_mediumage_female',
       'mean_salary_mediumage_male', 'mean_salary_oldage',
       'mean_salary_oldage_female', 'mean_salary_oldage_male'],
      dtype='object')
```

```
print(woman_managerial_position.columns)
print(french_cat_job.columns)
```
✓ 0.0s

```
Index(['time_period', '%_women_in_managerial_positions',
       '%_women_in_senior_middle_management', '%_women_in_junior_management'],
      dtype='object')
Index(['cat_job_id', 'french_cat_job', 'sub_cat_job_id', 'french_sub_cat_job'], dtype='object')
```

To summarize :

1. **Population Dataset** : population = 'population_clean.csv'
   - 'CODGEO' represents the geographical code
   - 'LIBGEO' represents the geographical name
   - 'job_cat' represents the job category
   - 'gender' represents the gender
   - 'age_cat' represents the age category
   - 'total_population' represents the total population in each geographical area

   ⇒ *population_all = 'population_final.csv':*
   - *population df before age categorization*
   - *includes the additional columns 'age' compared to the **population** dataset representing the age of the population by bins of 5 year*

2. **Salary Dataset** : salary = 'salary_concat.csv'
   - 'CODGEO' represents the geographical code
   - 'mean_salary' represents the mean salary
   - Columns like 'mean_salary_executive', 'mean_salary_middlemanagement', etc., represent the mean salary for different job categories
   - 'gender' represents the gender

⇒ *salary_all = 'salary_final.csv':*
- *salary df before normalization*
- *includes several additional columns compared to the **salary** dataset, which provide more detailed information about salaries based on gender and job category.*

3. **Establishment Dataset** : establishment = 'establishment_clean.csv'
   - 'CODGEO' represents the geographical code
   - 'total_establishment' represents the total number of establishments
   - Other columns represent the number of establishments based on their size and sector

4. **Geography Dataset** : geography = 'geography_clean.csv'
   - 'CODGEO' represents the geographical code.
   - 'postcode' represents the postal code
   - 'town_fullname' and 'town_name' represent the name of the town.
   - 'latitude' and 'longitude' represent the geographical coordinates
   - 'code_departement', 'departement_name', 'code_region', and 'region_name' represent the administrative divisions

5. **Proportion of women in managerial positions Dataset :**
   - woman_managerial_position='data_from_api_clean.csv'
   - time_period: Time period for the data
   - %_women_in_managerial_positions: Percentage of women in managerial positions
   - %_women_in_senior_middle_management: Percentage of women in senior middle management positions
   - %_women_in_junior_management: Percentage of women in junior management positions

6. **French job classification and socio-professional categories Dataset :**
   - french_cat_job **=** 'cat_job_webscrap.csv'
   - cat_job_id: Identifier for the job category
   - french_cat_job: Main job category in French
   - sub_cat_job_id: Identifier for the sub-category job
   - french_sub_cat_job: Sub-category job within the main job category in French

3. Exploratory data analysis & Visualization

A. EDA

- Checking number of observations

```
print(population.shape)
print(salary.shape)
print(establishment.shape)
print(establishment.shape)
```

✓ 0.0s

```
(826266, 6)
(10842, 10)
(34980, 11)
(34980, 11)
```

```
print(population_all.shape)
print(salary_all.shape)
```

✓ 0.0s

```
(1738965, 7)
(5421, 25)
```

```
print(woman_managerial_position.shape)
print(french_cat_job.shape)
```

✓ 0.0s

```
(19, 4)
(42, 4)
```

- Checking info of the data frames :

```
establishment.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34980 entries, 0 to 34979
Data columns (total 11 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   CODGEO                  34980 non-null  object
 1   total_establishment     34980 non-null  int64 
 2   micro_firms             34980 non-null  int64 
 3   small_firms             34980 non-null  int64 
 4   medium_firms            34980 non-null  int64 
 5   large_firms             34980 non-null  int64 
 6   agriculture_est         34980 non-null  int64 
 7   industry_est            34980 non-null  int64 
 8   construction_est        34980 non-null  int64 
 9   commerce_transport_est  34980 non-null  int64 
 10  public_est              34980 non-null  int64 
dtypes: int64(10), object(1)
memory usage: 2.9+ MB
```

- Descriptive statistics :

```
salary_by_gender = salary.groupby('gender').describe().round(2)

salary_by_gender
```

✓ 0.0s Open 'salary_by_gender'

	mean_salary								mean_salary_executive	
	count	mean	std	min	25%	50%	75%	max	count	mean
gender										
female	5421.0	13.82	2.34	10.40	12.36	13.18	14.54	34.60	5421.0	22.04
male	5421.0	16.60	3.88	10.78	14.28	15.65	17.69	60.48	5421.0	26.77

2 rows × 64 columns

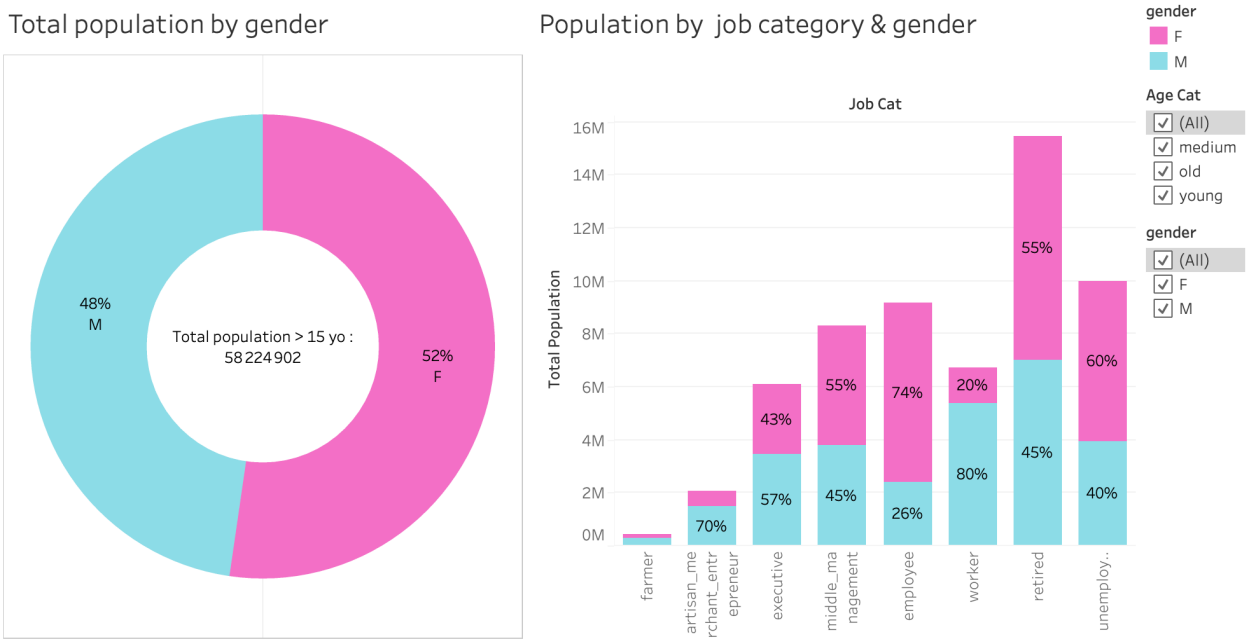
```
establishment_by_town = establishment.describe().round(2)
establishment_by_town
```

✓ 0.0s Open 'establishment_by_town'

	total_establishment	micro_firms	small_firms	medium_firms	large_firms	agriculture_est	industry_est	construction_est	commerce_transport_est	public_est
count	34980.00	34980.00	34980.00	34980.00	34980.00	34980.00	34980.00	34980.00	34980.00	34980.00
mean	64.60	46.90	9.25	1.93	0.39	3.10	4.24	7.03	41.87	8.36
std	386.04	279.72	56.35	12.20	3.38	5.58	16.03	29.79	307.67	41.93
min	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
25%	5.00	4.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00	1.00
50%	11.00	9.00	1.00	0.00	0.00	2.00	1.00	2.00	3.00	2.00
75%	29.00	22.00	4.00	1.00	0.00	4.00	3.00	5.00	13.00	4.00
max	21566.00	15633.00	3132.00	648.00	187.00	189.00	751.00	1750.00	19539.00	2163.00

B. Visualization

- Population & Establishment demographics :



We used two different measures jointly for the proportion of females in the total number of persons employed in managerial positions : the share of females in (total) management and the share of females in senior and middle management (thus excluding junior management). The joint calculation of these two measures provides information on whether women are more represented in junior management than in senior and middle management, thus pointing to an eventual ceiling for women to access higher-level management positions.

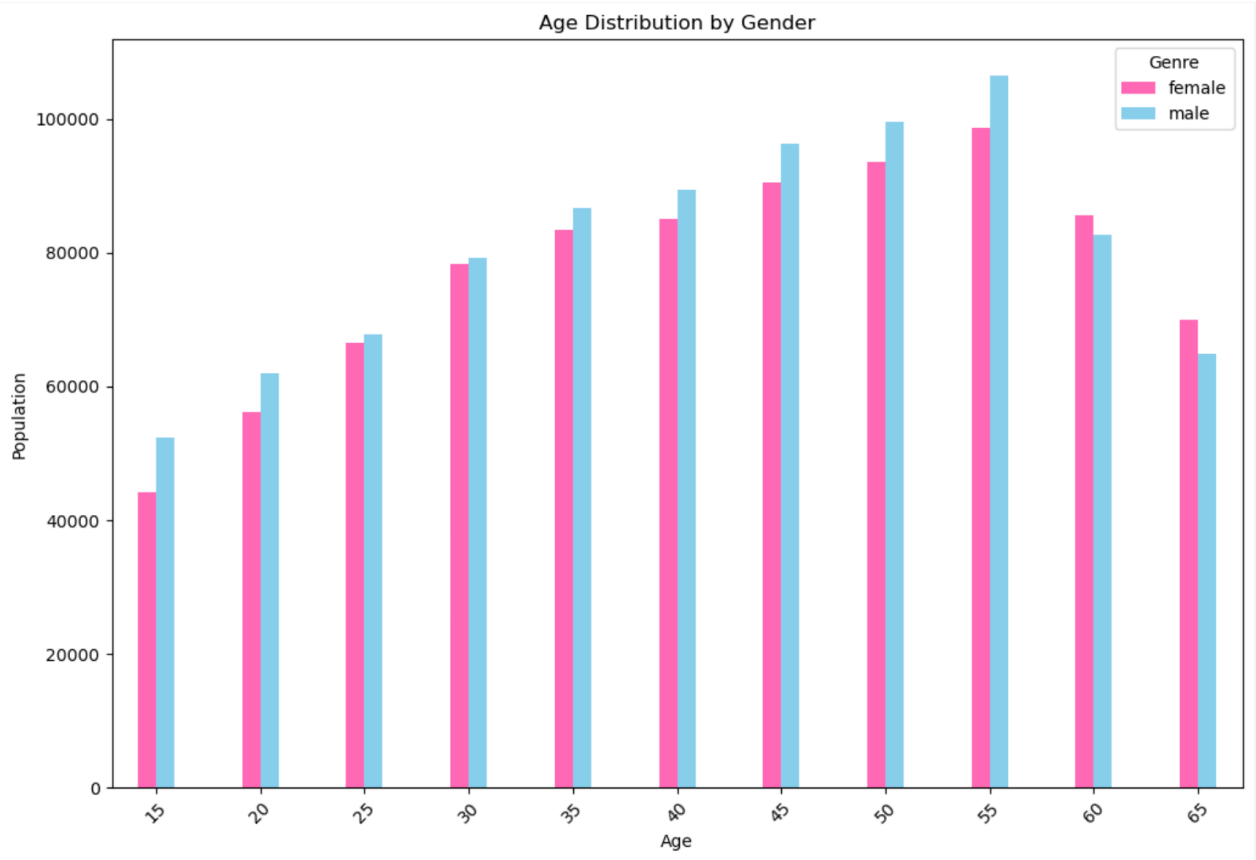
From our datasets, we can see that executive males (57%) are represented more than executive females (43%), even though we can see that there is more mid-manager female (55%) than mid-manager males (45%).

We can look at our proportion of women in managerial positions Dataset to see the evolution through years :

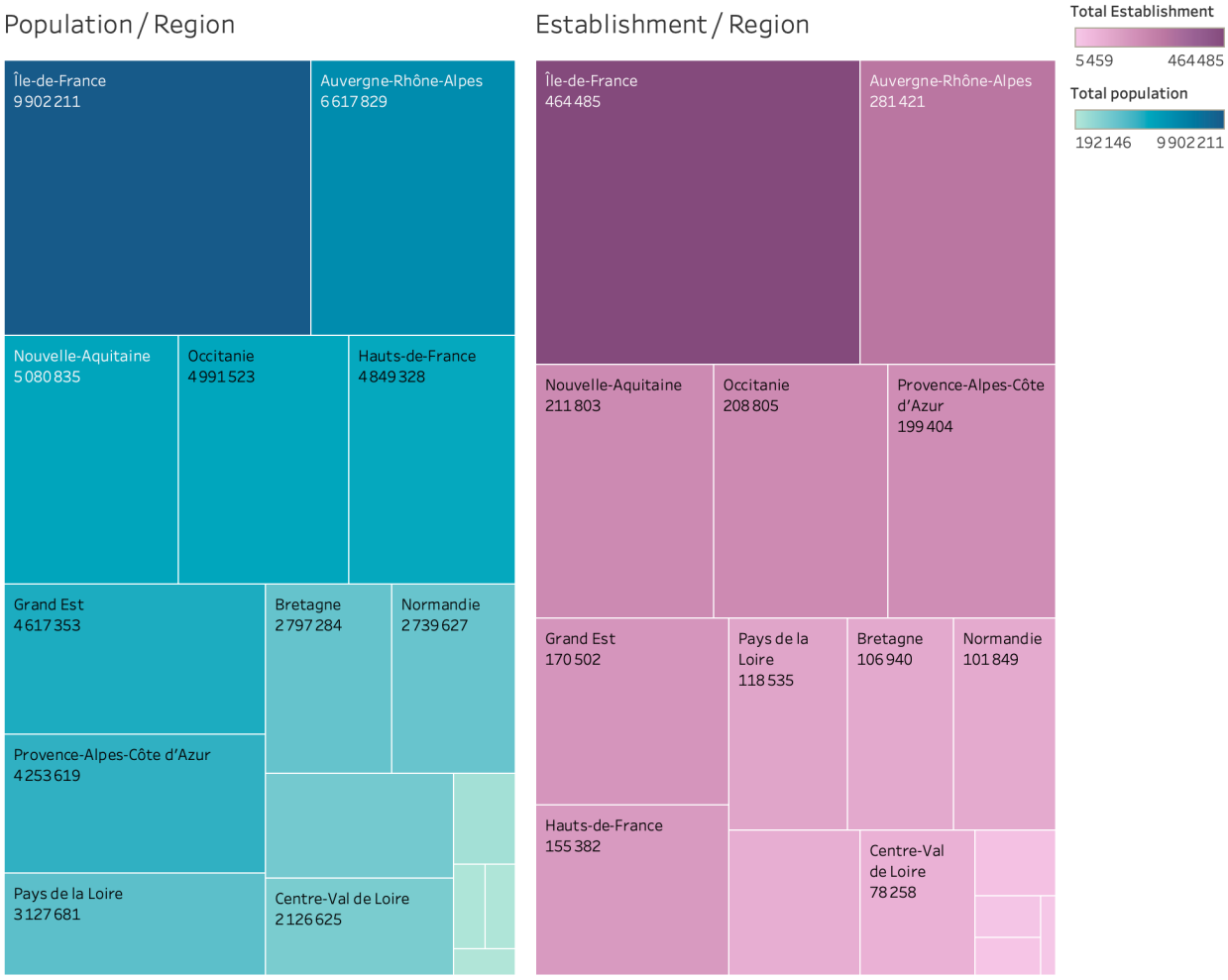
Proportion of woman in a managerial position

Year of Time Period	% Women In Managerial Positions	% Women In Senior Middle Management
2004	35,89	33,46
2005	37,60	36,37
2006	37,88	37,01
2007	37,83	36,46
2008	38,50	36,79
2009	38,09	37,37
2010	38,54	38,59
2011	39,34	39,13
2012	39,32	39,20
2013	36,00	35,58
2014	32,74	32,03
2015	31,66	30,93
2016	32,91	31,07
2017	33,44	32,64
2018	34,44	34,38
2019	34,67	34,30
2020	35,53	34,95
2021	37,79	36,77
2022	39,89	39,39

- Age distribution :



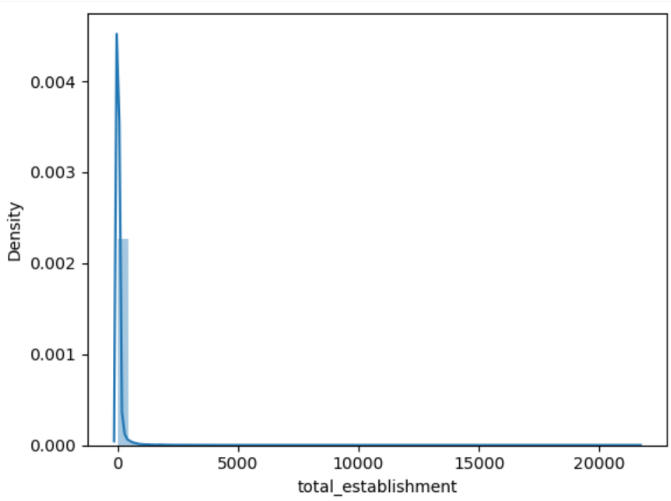
- Population & Establishment correlation :



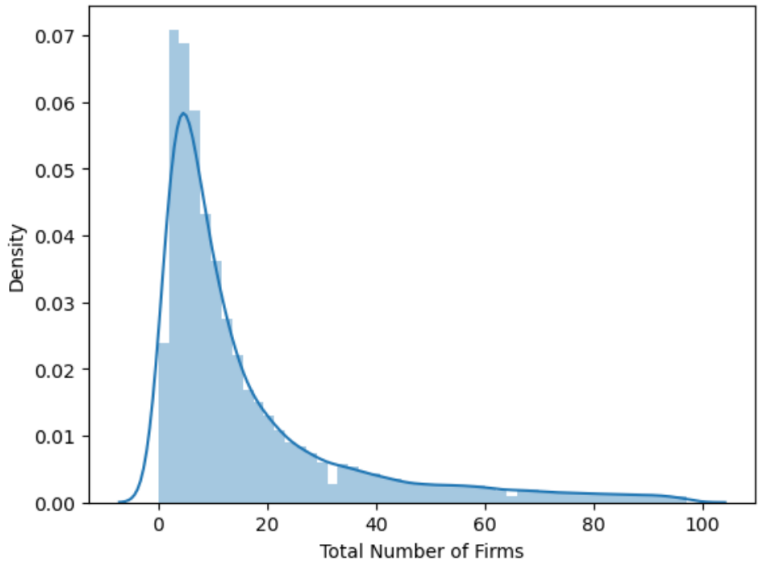
We can see that the most populated regions are also where there is the most establishment. The Pearson Correlation calculated is 0.6 between those two.

population_establishment_correlation
0.59

- Distribution of total number of establishments in town :

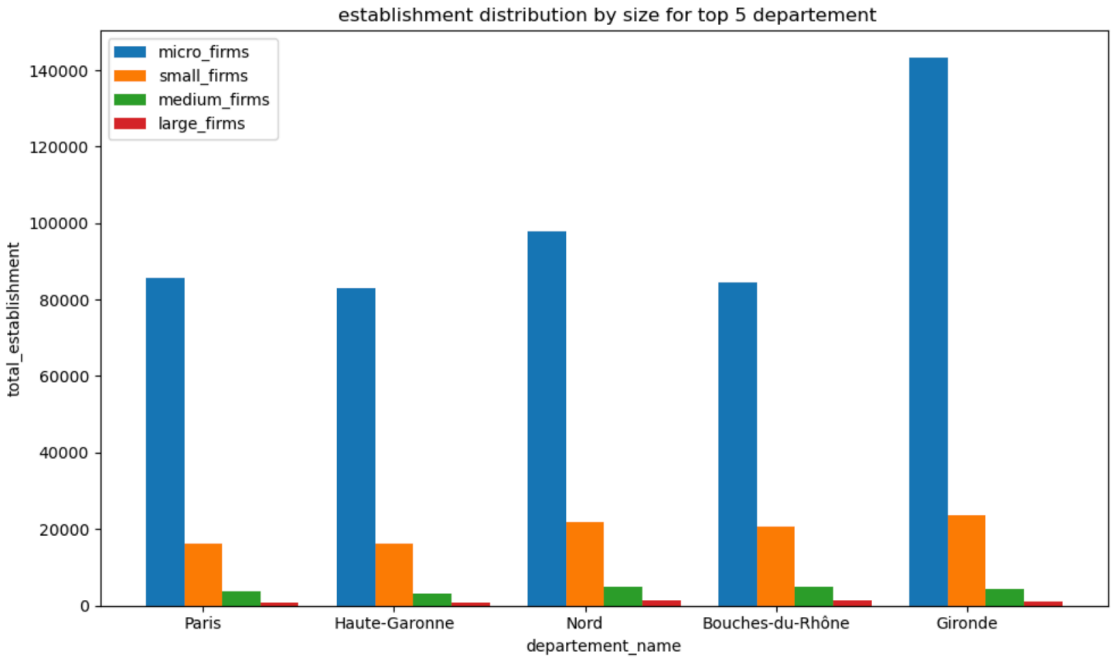


We can find that most of firms are not in a very big scale, so we need to focus on the establishments with relative low number.



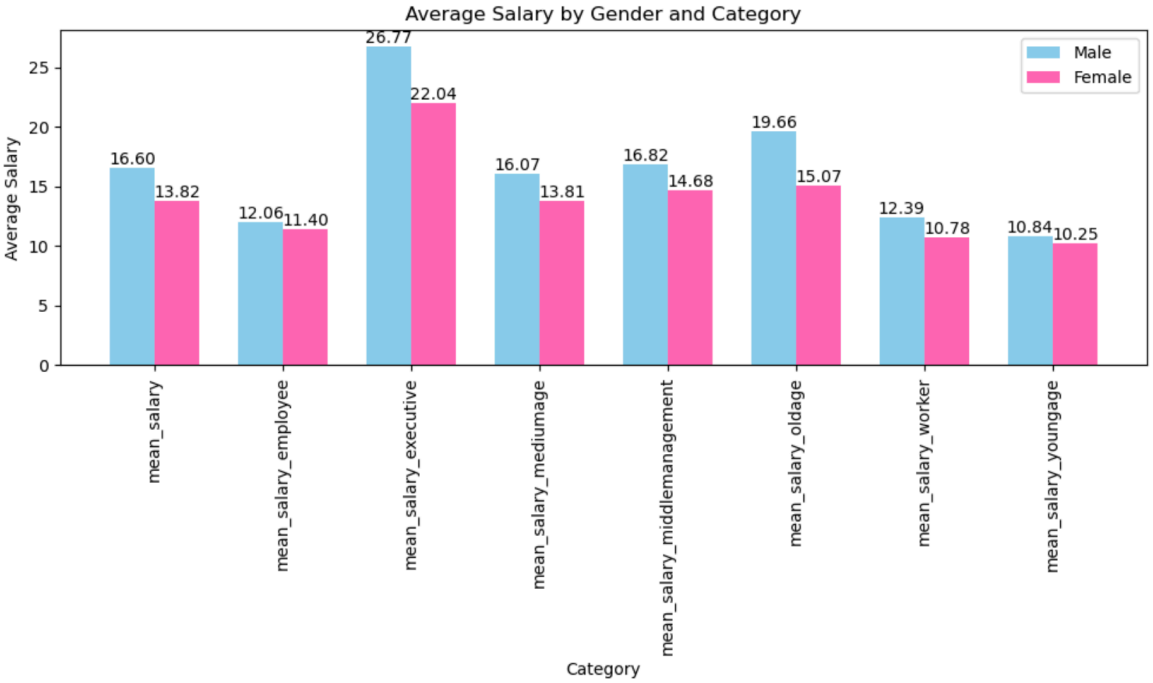
Most of firms' scale are 'Small' and 'Medium'.

Here is the distribution of establishment by size fort the top 5 departments with the most total establishments :

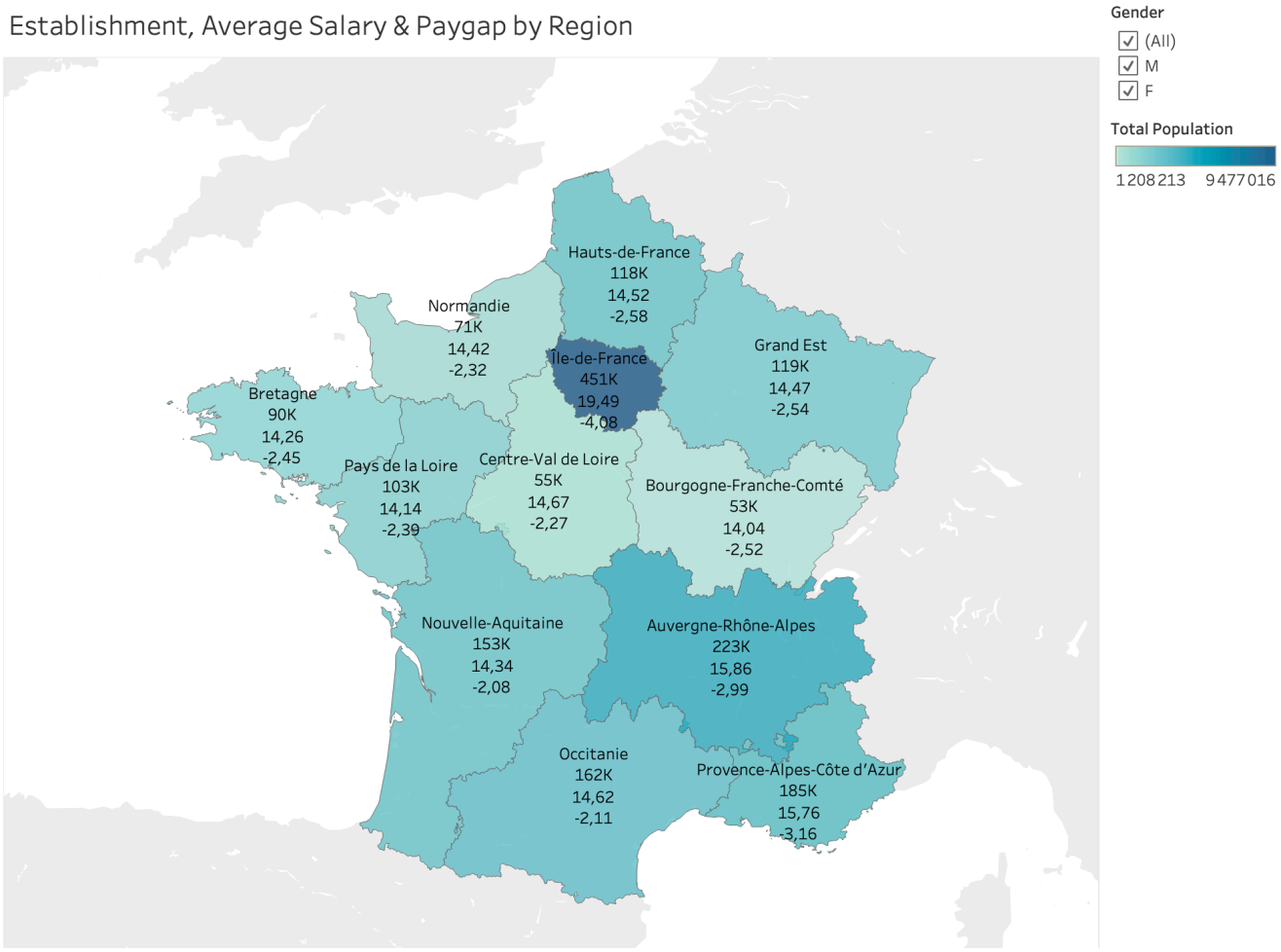


- Salary demographics :

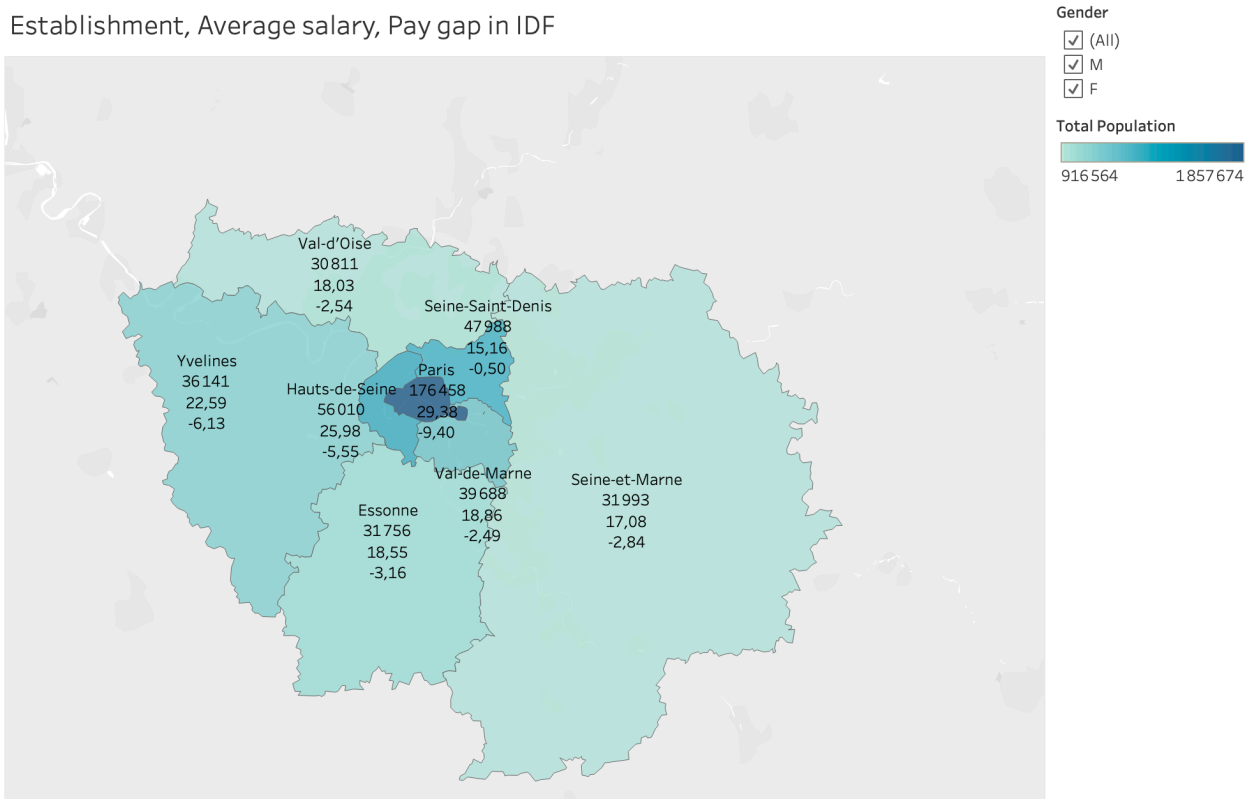
The gender inequalities are even higher in terms of salaries in the managerial positions, and the same goes with the age. The older the population is, the greater is the pay gap between male and female :



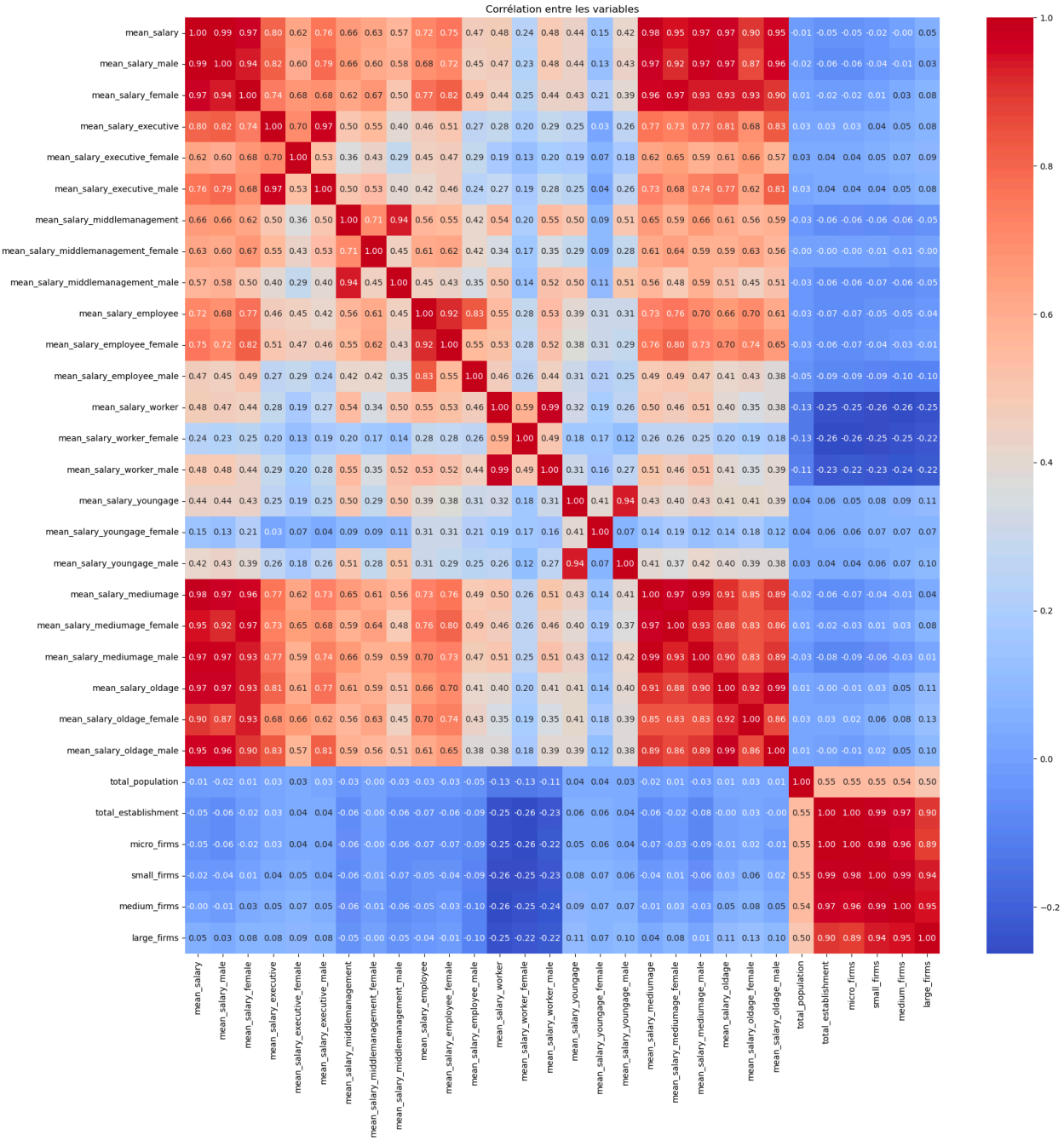
Regions most populated have a higher salary, but also a higher gender pay gap :



Here is a focus on Ile-de-France :



To examine the correlations between our several features (gender, establishment size, population, salaries and job categories, we did a heatmap :



We can see that population is correlated with the firm size and salaries are correlated with gender and job categories. But salaries are not correlated to firm sizes nor population.

III. Database, ERD, SQL

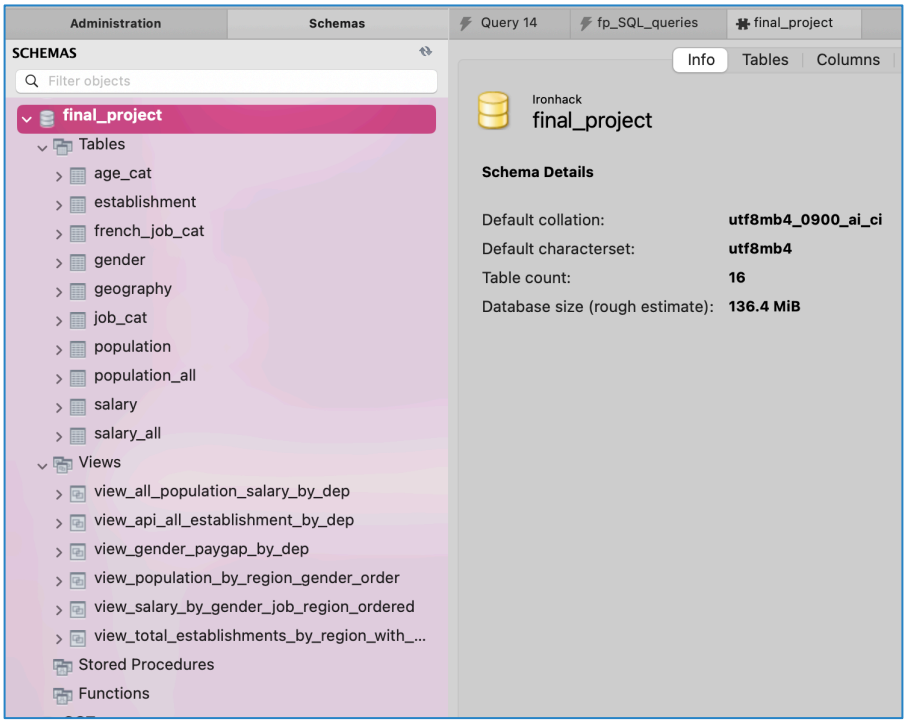
1. Database

A. Type of selection

When it comes to the choice of a database type, since we have structured data organized into interrelated tables with a predefined schema, relationships, and foreign keys it seemed appropriate to use a Relational Database. This type of database will help to reduce data redundancy and improve data integrity, easily manipulate data, and then, use SQL to perform queries that involve joining data from multiple tables.

B. Database creation

The 'final_project' database was created in MySQL Workbench to store 7 tables related to our subject which we collected through flat files, API and web scraping, and 3 tables that we have normalized from our datasets.




```
salary.to_sql('salary',engine, 'final_project', if_exists='replace', index=False)
salary_all.to_sql('salary_all',engine, 'final_project', if_exists='replace', index=False)
population.to_sql('population',engine, 'final_project', if_exists='replace', index=False)
population_all.to_sql('population_all',engine, 'final_project', if_exists='replace', index=False)
geography.to_sql('geography',engine, 'final_project', if_exists='replace', index=False)
establishment.to_sql('establishment',engine, 'final_project', if_exists='replace', index=False)
french_job_cat.to_sql('french_job_cat',engine, 'final_project', if_exists='replace', index=False)

✓ 0.1s
```

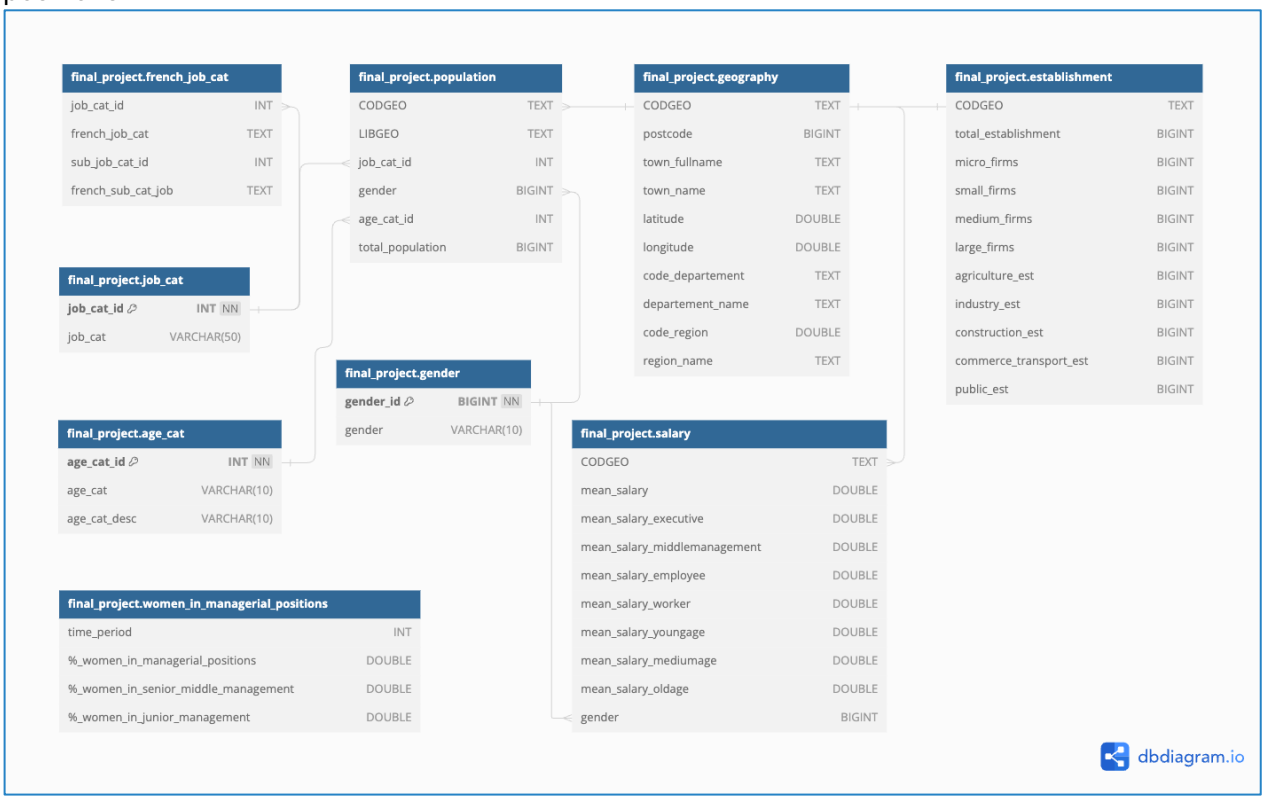
Query 14 | fp_SQL_queries | final_project

Info | Tables | Columns | Indexes | Triggers | Views | Stored Procedures | Functions | Grants | Events

Name	Engine	Version	Row Format	Rows	Avg Row Length	Data Length	Max Data Length	Index Length	Data F
age_cat	InnoDB	10	Dynamic	2	8192	16.0 KiB	0.0 bytes	0.0 bytes	0.
establishment	InnoDB	10	Dynamic	34840	135	4.5 MiB	0.0 bytes	0.0 bytes	
french_job_cat	InnoDB	10	Dynamic	42	390	16.0 KiB	0.0 bytes	0.0 bytes	0.
gender	InnoDB	10	Dynamic	2	8192	16.0 KiB	0.0 bytes	0.0 bytes	0.
geography	InnoDB	10	Dynamic	35349	133	4.5 MiB	0.0 bytes	0.0 bytes	
job_cat	InnoDB	10	Dynamic	8	2048	16.0 KiB	0.0 bytes	0.0 bytes	0.
population	InnoDB	10	Dynamic	823557	77	60.6 MiB	0.0 bytes	48.2 MiB	1
population_all	InnoDB	10	Dynamic	228508	71	15.5 MiB	0.0 bytes	0.0 bytes	0.
salary	InnoDB	10	Dynamic	10944	145	1.5 MiB	0.0 bytes	0.0 bytes	
salary_all	InnoDB	10	Dynamic	5252	302	1.5 MiB	0.0 bytes	0.0 bytes	

C. Entity-Relationship Diagram (ERD)

Our Entity-Relationship Diagram (ERD) provides a visual representation of the database schema, illustrating the relationships between different entities within the database. In our project's ERD, we have several tables representing various aspects of the data, including population demographics, salary information, geographical data, job categories, and women's representation in managerial positions.



D. Insights from the data collected through mySQL

Using MySQL queries to gather insights from the data collected :

a. Query 1 – Pearson correlation between establishment and population for EDA

```sql
-- Pearson correlation establishment and population
SELECT ROUND((
      COUNT(*) * SUM(p.total_population * e.total_establishment) -
      SUM(p.total_population) * SUM(e.total_establishment)
    ) / (
      SQRT((COUNT(*) * SUM(p.total_population * p.total_population)) - (SUM(p.total_population) * SUM(p.total_population))) *
      SQRT((COUNT(*) * SUM(e.total_establishment * e.total_establishment)) - (SUM(e.total_establishment) * SUM(e.total_establishment)))
    ), 2) AS population_establishment_correlation
FROM
    population p
JOIN
    geography ge ON p.CODGEO = ge.CODGEO
JOIN
    establishment e ON p.CODGEO = e.CODGEO;
```

Result :

population_establishment_correlation
0.59

As we said in the EDA part, the number of establishments and the total population is correlated.

b. Query 2 – View of the average gender pay gap by department to integrate in Tableau

```sql
CREATE VIEW view_gender_paygap_by_dep AS
SELECT
    ge.code_departement,
    ge.departement_name,
    ROUND(AVG(CASE WHEN s.gender = '1' THEN s.mean_salary END),2) AS avg_salary_male,
    ROUND(AVG(CASE WHEN s.gender = '2' THEN s.mean_salary END),2) AS avg_salary_female,
    ROUND(AVG(s.mean_salary),2) AS avg_salary_all,
    ROUND(AVG(CASE WHEN s.gender = '2' THEN s.mean_salary END)-AVG(CASE WHEN s.gender = '1' THEN s.mean_salary END),2) AS gender_pay_gap
FROM
    salary s
JOIN geography ge ON s.CODGEO = ge.CODGEO
GROUP BY ge.code_departement, ge.departement_name
UNION
SELECT
    NULL AS code_departement,
    'Total' AS departement_name,
    ROUND(AVG(CASE WHEN s.gender = '1' THEN s.mean_salary END), 2) AS avg_salary_male,
    ROUND(AVG(CASE WHEN s.gender = '2' THEN s.mean_salary END), 2) AS avg_salary_female,
    ROUND(AVG(s.mean_salary), 2) AS avg_salary_all,
    ROUND(AVG(CASE WHEN s.gender = '2' THEN s.mean_salary END) - AVG(CASE WHEN s.gender = '1' THEN s.mean_salary END), 2) AS gender_pay_gap
FROM
    salary s;
SELECT * FROM view_gender_paygap_by_dep;
```

Result :

code_departem...	departement_na...	avg_salary_male	avg_salary_female	avg_salary_all	gender_pay_gap
10	Aube	14.61	12.45	13.53	-2.16
11	Aude	14.01	12.25	13.13	-1.75
12	Aveyron	14.42	12.42	13.42	-1.99
13	Bouches-du-Rhône	18.53	14.76	16.64	-3.76
14	Calvados	15.54	13.15	14.34	-2.38
15	Cantal	13.87	12.17	13.02	-1.7
16	Charente	15.1	12.94	14.02	-2.16
17	Charente-Maritime	15.7	13.02	14.36	-2.68
18	Cher	14.86	12.55	13.71	-2.31
19	Corrèze	14.58	12.66	13.62	-1.93
21	Côte-d'Or	15.44	13.14	14.29	-2.29
22	Côtes-d'Armor	15.07	12.81	13.94	-2.25
23	Creuse	12.82	12.14	12.48	-0.68
24	Dordogne	13.98	12.4	13.19	-1.58

We have exported this view as a CSV to import it on Tableau for our visualization on gender pay gap.

c. *Query 3 – View of population and salary data by department and gender to integrate to our API for the 1st endpoint*

```sql
CREATE VIEW view_all_population_salary_by_dep AS
SELECT
    ge.code_departement,
    ge.departement_name,
    g.gender,
    ROUND(SUM(p.total_population), 2) AS total_population,
    ROUND(AVG(s.mean_salary), 2) AS average_salary,
    ROUND(SUM(CASE WHEN p.age_cat_id = '1' THEN p.total_population ELSE 0 END), 2) AS total_population_under25yo,
    ROUND(AVG(s.mean_salary_youngage), 2) AS average_salary_under25yo,
    ROUND(SUM(CASE WHEN p.age_cat_id = '2' THEN p.total_population ELSE 0 END), 2) AS total_population_25to49yo,
    ROUND(AVG(s.mean_salary_mediumage), 2) AS average_salary_25to49yo,
    ROUND(SUM(CASE WHEN p.age_cat_id = '3' THEN p.total_population ELSE 0 END), 2) AS total_population_over50yo,
    ROUND(AVG(s.mean_salary_oldage), 2) AS average_salary_over50yo,
    ROUND(SUM(CASE WHEN j.job_cat = 'executive' THEN p.total_population ELSE 0 END), 2) AS total_population_executive,
    ROUND(AVG(s.mean_salary_executive), 2) AS average_salary_executive,
    ROUND(SUM(CASE WHEN j.job_cat = 'middle_management' THEN p.total_population ELSE 0 END), 2) AS total_population_middle_management,
    ROUND(AVG(s.mean_salary_middlemanagement), 2) AS average_salary_middlemanagement,
    ROUND(SUM(CASE WHEN j.job_cat = 'employee' THEN p.total_population ELSE 0 END), 2) AS total_population_employee,
    ROUND(AVG(s.mean_salary_employee), 2) AS average_salary_employee,
    ROUND(SUM(CASE WHEN j.job_cat = 'worker' THEN p.total_population ELSE 0 END), 2) AS total_population_worker,
    ROUND(AVG(s.mean_salary_worker), 2) AS average_salary_worker
FROM population p
JOIN gender g ON p.gender = g.gender_id
JOIN job_cat j ON j.job_cat_id = p.job_cat_id
JOIN salary s ON p.CODGEO = s.CODGEO AND p.gender = s.gender
JOIN geography ge ON p.CODGEO = ge.CODGEO
GROUP BY ge.code_departement, ge.departement_name, g.gender
ORDER BY
    ge.code_departement,
    FIELD(g.gender, 'male', 'female');
SELECT * FROM view_all_population_salary_by_dep;
```

Result :

code_departement	departement_na...	gender	total_populati...	average_salary	total_population_under25yo	average_salary_under25...	total_population_25to4...	averag
10	Aube	Male	68967	14.62	12914	10.44	25421	14.33
10	Aube	Female	78780	12.45	11570	9.94	27207	12.52
11	Aude	Male	84188	13.97	11344	10.01	28572	13.61
11	Aude	Female	96729	12.24	11120	9.89	30312	12.17
12	Aveyron	Male	62637	14.41	8974	10.39	21650	14.14
12	Aveyron	Female	68047	12.42	7713	10	21352	12.31
13	Bouches-du-Rhône	Male	1142925	18.44	189083	11.13	440071	17.73
13	Bouches-du-Rhône	Female	1293692	14.71	193637	10.36	465531	14.8
14	Calvados	Male	192292	15.48	34245	10.46	69209	15.08
14	Calvados	Female	220585	13.09	34386	10.04	70709	13.09
15	Cantal	Male	23834	13.85	3230	10.27	8079	13.55
15	Cantal	Female	27568	12.15	3297	9.77	8063	12.19
16	Charente	Male	74496	15.1	11353	10.49	26181	14.55
16	Charente	Female	85314	12.95	10212	10.16	27261	12.9

We have used this view for creating our API to expose our data.

d. *Query 4 – salaries, total population, paygap, correlation and firms data by department to integrate in our API for the second endpoint*

```sql
-- pour API - pop_est_correlation_and_paygap
SELECT
    vall.code_departement,
    vall.departement_name,
    SUM(vall.total_population) AS total_population,
    ROUND(AVG(pg.avg_salary_all),2) AS average_salary_all,
    ROUND(AVG(pg.avg_salary_male),2) AS average_salary_male,
    ROUND(AVG(pg.avg_salary_female),2) AS average_salary_female,
    ROUND(AVG(pg.gender_pay_gap),2) AS gender_paygap,
    ve.population_establishment_correlation,
    SUM(e.total_establishment) AS total_establishments,
    SUM(e.micro_firms) AS total_micro_firms,
    SUM(e.small_firms) AS total_small_firms,
    SUM(e.medium_firms) AS total_medium_firms,
    SUM(e.large_firms) AS total_large_firms
FROM view_all_population_salary_by_dep vall
JOIN view_gender_paygap_by_dep pg ON vall.code_departement = pg.code_departement
JOIN geography ge ON ge.code_departement = vall.code_departement
JOIN view_api_all_establishment_by_dep ve ON ve.code_departement = vall.code_departement
JOIN establishment e ON ge.CODGEO = e.CODGEO
GROUP BY vall.code_departement, vall.departement_name, ve.population_establishment_correlation;
```

Result :

code_departem...	departement_na...	total_population	average_salary_all	average_salary_male	average_salary_female	gender_paygap	population_establishment_correla...	total_estab
12	Aveyron	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
13	Bouches-du-Rhône	353309465	16.64	18.53	14.76	-3.76	0.51	235580
14	Calvados	220476318	14.34	15.54	13.15	-2.38	0.57	49822
15	Cantal	12696294	13.02	13.87	12.17	-1.7	0.48	11414
16	Charente	58170840	14.02	15.1	12.94	-2.16	0.54	24202
17	Charente-Maritime	165042369	14.36	15.7	13.02	-2.68	0.5	50296
18	Cher	40461984	13.71	14.86	12.55	-2.31	0.54	18326
19	Corrèze	29700108	13.62	14.58	12.66	-1.93	0.51	16706
21	Côte-d'Or	190267820	14.29	15.44	13.14	-2.29	0.61	36656
22	Côtes-d'Armor	116903134	13.94	15.07	12.81	-2.25	0.46	39642
23	Creuse	5723904	12.48	12.82	12.14	-0.68	0.45	8082
24	Dordogne	84279450	13.19	13.98	12.4	-1.58	0.47	30722
25	Doubs	157142385	14.36	15.81	12.91	-2.9	0.61	33262
26	Drôme	107813904	14.69	16.03	13.34	-2.69	0.56	39108
27	Eure	158377254	14.64	15.79	13.48	-2.31	0.57	34842
28	Eure-et-Loir	74220448	14.77	15.95	13.59	-2.36	0.54	25238

e. *Query 5 & 6 – TOP 10 department with higher and lowest gender pay gap for EDA*

```sql
select *
 from view_gender_paygap_by_dep
 order by gender_pay_gap ASC limit 10 ;

 select *
 from view_gender_paygap_by_dep
 order by gender_pay_gap DESC limit 10 ;
```

Result :

- Department with highest pay gap :

code_departem...	departement_na...	avg_salary_male	avg_salary_female	avg_salary_all	gender_pay_gap
75	Paris	34.67	25.27	29.97	-9.4
78	Yvelines	25.65	19.52	22.59	-6.13
92	Hauts-de-Seine	28.78	23.22	26	-5.55
69	Rhône	20.43	16.16	18.3	-4.27
90	Territoire de Belfort	17.83	13.73	15.78	-4.11
31	Haute-Garonne	18.75	14.99	16.87	-3.76
13	Bouches-du-Rhône	18.53	14.76	16.64	-3.76
38	Isère	18.47	14.82	16.65	-3.65
73	Savoie	17.26	13.96	15.61	-3.3
68	Haut-Rhin	16.5	13.28	14.89	-3.22

- Department with lowest pay gap :

code_departem...	departement_na...	avg_salary_male	avg_salary_female	avg_salary_all	gender_pay_gap
93	Seine-Saint-Denis	15.38	14.88	15.13	-0.5
23	Creuse	12.82	12.14	12.48	-0.68
48	Lozère	13.15	12.16	12.65	-0.99
974	La Réunion	13.89	12.77	13.33	-1.12
971	Guadeloupe	14.69	13.56	14.12	-1.12
973	Guyane	15.09	13.96	14.52	-1.13
79	Deux-Sèvres	14.64	13.19	13.91	-1.45
972	Martinique	15.09	13.64	14.36	-1.45
36	Indre	13.82	12.31	13.06	-1.51
24	Dordogne	13.98	12.4	13.19	-1.58

We can see that even the lowest pay gap is still negative. Meaning that the pay gap between female and male is always negative for all departments of France. Women earn less than men everywhere, and gender inequalities is at this highest in the capital at Paris, Ile-de-France.

IV. API

To expose portion of data from the database I have created an API that allows users, to retrieve specific data on salaries, population by gender, and firms across French department. It provides a valuable resource for analyzing gender disparities and inequalities in France in 2020.

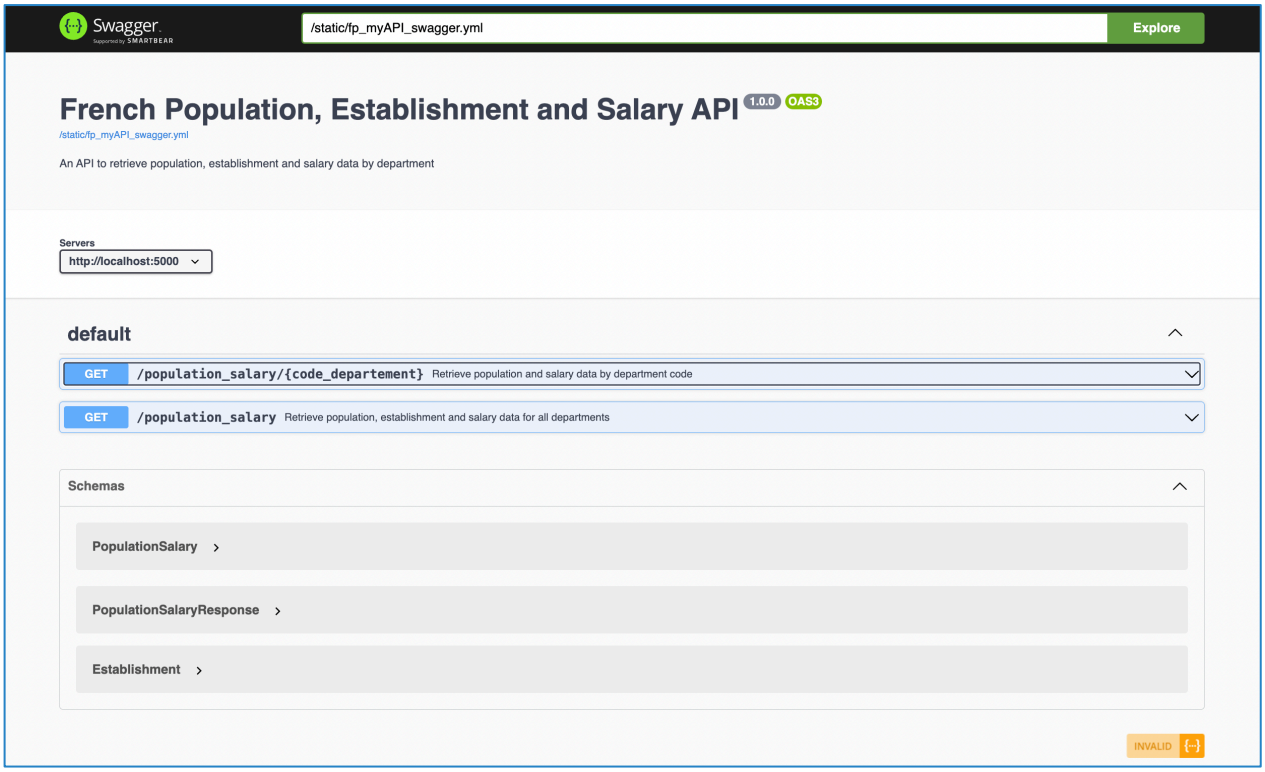
The API has been built using Flask and provides responses in JSON format.

main > fp_myAPI.py > ...

```
12  swaggerui_blueprint = get_swaggerui_blueprint(
13      base_url='/docs',
14      api_url='/static/fp_myAPI_swagger.yml',
15  )
16  app = Flask(__name__)
17  app.register_blueprint(swaggerui_blueprint)
18
19  def remove_null_fields(obj):
20      return {k:v for k, v in obj.items() if v is not None}
21
22
23  @app.route("/population_salary/<code_departement>")
24
25  def population_salary_by_dep(code_departement):
26      db_conn = pymysql.connect(host="localhost"
27                                , user="root"
28                                ,  password=pw_raw
29                                , database="final_project",
30                                cursorclass=pymysql.cursors.DictCursor)
31      with db_conn.cursor() as cursor:
32          cursor.execute("""
33                      SELECT *
34                      FROM view_all_population_salary_by_dep
35                      WHERE code_departement = %s
36                      """,
37                      (code_departement,)
38                          )
39          population_salary_by_dep = cursor.fetchall()
40
```

1. Swagger

http://192.168.1.18:8080/docs/ :



2. Endpoints

A. Endpoint 1 : /population_salary

http://192.168.1.18:8080/population_salary :


```
[
  {
    "average_salary_all": 13.53,
    "average_salary_female": 12.45,
    "average_salary_male": 14.61,
    "code_departement": "10",
    "departement_name": "Aube",
    "gender_paygap": -2.16,
    "population_establishment_correlation": 0.58,
    "total_establishments": "687238",
    "total_population": "515460"
  },
  {
    "average_salary_all": 13.13,
    "average_salary_female": 12.25,
    "average_salary_male": 14.01,
    "code_departement": "11",
    "departement_name": "Aude",
    "gender_paygap": -1.75,
    "population_establishment_correlation": 0.53,
    "total_establishments": "967828",
    "total_population": "630998"
  },
  {
    "average_salary_all": 13.42,
    "average_salary_female": 12.42,
    "average_salary_male": 14.42,
    "code_departement": "12",
    "departement_name": "Aveyron",
    "gender_paygap": -1.99,
    "population_establishment_correlation": 0.49,
    "total_establishments": "856544",
    "total_population": "507702"
  },
  {
    "average_salary_all": 16.64,
    "average_salary_female": 14.76,
    "average_salary_male": 18.53,
    "code_departement": "13",
    "departement_name": "Bouches-du-Rhône",
    "gender_paygap": -3.76,
    "population_establishment_correlation": 0.51,
    "total_establishments": "10158796",
    "total_population": "4903278"
  },
  {
    "average_salary_all": 14.34,
    "average_salary_female": 13.15,
    "average_salary_male": 15.54,
    "code_departement": "14",
    "departement_name": "Calvados",
    "gender_paygap": -2.38,
    "population_establishment_correlation": 0.57,
    "total_establishments": "1795902",
    "total_population": "1209556"
  },
  {
    "average_salary_all": 13.02,
    "average_salary_female": 12.17,
    "average_salary_male": 13.87,
    "code_departement": "15",
```

B. Endpoint 2 : /population_salary/<departement_code>

http://192.168.1.18:8080/population_salary/75 :


```
"establishment": [
    {
        "code_departement": "75",
        "departement_name": "Paris",
        "population_establishment_correlation": 0.08,
        "total_agriculture_sector": "5841",
        "total_commerce_transport_sector": "7067273",
        "total_construction_sector": "347680",
        "total_establishments": "8186115",
        "total_industry_sector": "207716",
        "total_large_firms": "47007",
        "total_medium_firms": "179133",
        "total_micro_firms": "6138983",
        "total_population": "1997602",
        "total_public_sector": "557605",
        "total_small_firms": "1004958"
    }
],
"population_salary": [
    {
        "average_salary": 34.58,
        "average_salary_25to49yo": 31.8,
        "average_salary_employee": 13.47,
        "average_salary_executive": 44.77,
        "average_salary_middlemanagement": 20.68,
        "average_salary_over50yo": 46.87,
        "average_salary_under25yo": 14.01,
        "average_salary_worker": 11.95,
        "code_departement": "75",
        "departement_name": "Paris",
        "gender": "Male",
        "total_population": "923801",
        "total_population_25to49yo": "426698",
        "total_population_employee": "82595",
        "total_population_executive": "311982",
        "total_population_middle_management": "118075",
        "total_population_over50yo": "354501",
        "total_population_under25yo": "142602",
        "total_population_worker": "59674"
    },
    {
        "average_salary": 25.27,
        "average_salary_25to49yo": 25.48,
        "average_salary_employee": 12.86,
        "average_salary_executive": 33.27,
        "average_salary_middlemanagement": 18.04,
        "average_salary_over50yo": 28.9,
        "average_salary_under25yo": 13.08,
        "average_salary_worker": 11.42,
        "code_departement": "75",
        "departement_name": "Paris",
        "gender": "Female",
        "total_population": "1073801",
        "total_population_25to49yo": "452013",
        "total_population_employee": "150780",
        "total_population_executive": "287006",
        "total_population_middle_management": "162726",
        "total_population_over50yo": "451805",
        "total_population_under25yo": "169983",
        "total_population_worker": "19714"
    }
```

Conclusion

In conclusion, this project represents a comprehensive exploration of the socio-economic landscape of France through the lens of extensive data analysis. By leveraging datasets from reputable sources such as the French National Institute of Statistics and Economic Studies (INSEE) and employing advanced analytical techniques, we've uncovered valuable insights into population demographics, establishment distributions, salary structures, and job categories across French departments and regions.

Through our analysis, we've identified patterns, trends, and correlations that offer a deeper understanding of societal complexities and economic dynamics within France. Our exploration has highlighted persistent challenges such as gender wage disparities and the underrepresentation of women in managerial positions, underscoring the importance of addressing these issues to foster a more inclusive and equitable labor market.

By examining data on population demographics, establishment distributions, and salary structures, we've provided valuable insights to inform decision-making and drive socio-economic progress.

Furthermore, the development of an API to access specific data points underscores our commitment to transparency, accessibility, and data-driven decision-making. The API provides a user-friendly interface for retrieving essential information on salaries, population demographics, and firm distributions, empowering stakeholders to conduct further analysis and research to support evidence-based policymaking and advocacy efforts.

In essence, this project represents a significant step towards understanding and addressing socio-economic challenges in France. By harnessing the power of data and analytics, we've illuminated key insights that can drive positive change and pave the way for a more prosperous and inclusive future for all citizens.

GDPR

After a thorough assessment of the data collected for this project, I can confidently affirm that no personal data has been utilized in any part of the processes. The sources of the data are entirely public and at a country level, ensuring full transparency and accessibility. Therefore, this project adheres to the guidelines and principles of the General Data Protection Regulation (GDPR).

References

- Github repository : https://github.com/steprag/fp_finale_project
- Flat files sources :
    - https://www.insee.fr/fr/statistiques/4991205
    - https://www.insee.fr/fr/statistiques/2021266
    - https://www.insee.fr/fr/statistiques/7631680?sommaire=7632456
    - https://www.data.gouv.fr/fr/datasets/communes-de-france-base-des-codes-postaux/
- SDG API link : https://unstats.un.org/sdgapi/swagger/#!/Target/V1SdgTargetDataGet
- Target for the API : https://sdgs.un.org/goals/goal5#targets_and_indicators
- Webscraping source :
    - https://fr.wikipedia.org/wiki/Professions_et_cat%C3%A9gories_socioprofessionnelles_en_France