In [1]:
```
# Initial imports
import pandas as pd
```

In [2]:
```
# read scv file
file_path=("34100133-eng/34100133.csv")
rent_df=pd.read_csv(file_path)
rent_df.isnull().sum()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3028753424.py:3: DtypeWarning: Columns (14) have mixed ty
pes. Specify dtype option on import or set low_memory=False.
  rent_df=pd.read_csv(file_path)
```

Out[2]:
```
REF_DATE                  0
GEO                       0
DGUID                   884
Type of structure         0
Type of unit              0
UOM                       0
UOM_ID                    0
SCALAR_FACTOR             0
SCALAR_ID                 0
VECTOR                    0
COORDINATE                0
VALUE                 55782
STATUS                66578
SYMBOL               122360
TERMINATED           119808
DECIMALS                  0
dtype: int64
```

In [3]:
```
rent_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 122360 entries, 0 to 122359
Data columns (total 16 columns):
 #   Column             Non-Null Count   Dtype
---  ------             --------------   -----
 0   REF_DATE           122360 non-null  int64
 1   GEO                122360 non-null  object
 2   DGUID              121476 non-null  object
 3   Type of structure  122360 non-null  object
 4   Type of unit       122360 non-null  object
 5   UOM                122360 non-null  object
 6   UOM_ID             122360 non-null  int64
 7   SCALAR_FACTOR      122360 non-null  object
 8   SCALAR_ID          122360 non-null  int64
 9   VECTOR             122360 non-null  object
 10  COORDINATE         122360 non-null  object
 11  VALUE              66578 non-null   float64
 12  STATUS             55782 non-null   object
 13  SYMBOL             0 non-null       float64
 14  TERMINATED         2552 non-null    object
 15  DECIMALS           122360 non-null  int64
dtypes: float64(2), int64(4), object(10)
memory usage: 14.9+ MB
```

In [4]:
```
rent_df.head()
```

Out[4]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COORDI
0	1987	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Bachelor units	Dollars	81	units	0	v42135513	19
1	1987	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	One bedroom units	Dollars	81	units	0	v42135529	19
2	1987	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Two bedroom units	Dollars	81	units	0	v42135545	19
3	1987	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Three bedroom units	Dollars	81	units	0	v42135561	19
4	1987	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row structures of three units and over	Bachelor units	Dollars	81	units	0	v42135577	19

In [5]:
```
rent_df.tail()
```

Out[5]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COOR
122355	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of three units and over	Three bedroom units	Dollars	81	units	0	v3824416	
122356	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Bachelor units	Dollars	81	units	0	v3824602	
122357	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	One bedroom units	Dollars	81	units	0	v3824790	
122358	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Two bedroom units	Dollars	81	units	0	v3824978	
122359	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Three bedroom units	Dollars	81	units	0	v3825166	

In [6]:
```
# checking all the columns value to see if it contains meaningful data
rent_df['UOM'].unique()
# rent_df['UOM_ID'].unique()
# rent_df['SCALAR_FACTOR'].unique()
# rent_df['SCALAR_ID'].unique()
# rent_df['STATUS'].unique()
# rent_df['SYMBOL'].unique()
# rent_df['TERMINATED'].unique()
# rent_df['DECIMALS'].unique()
```

Out[6]:
```
array(['Dollars'], dtype=object)
```

In [7]:
```
# no useful data on these columns, therefor dropping them
columns_to_drop = ['UOM','UOM_ID', 'SCALAR_FACTOR', 'SCALAR_ID', 'STATUS','SYMBOL', 'TERMINATED','DECIMALS']
rent_df.drop(columns=columns_to_drop, inplace=True)
```

In [8]:
```
# last 10 years, dropping any data before 2012
rent_10years_df = rent_df[rent_df['REF_DATE'] >= 2012]
rent_10years_df.head()
```

Out[8]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE
85736	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Bachelor units	v42135513	192.3.1	NaN
85737	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	One bedroom units	v42135529	192.3.2	NaN
85738	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Two bedroom units	v42135545	192.3.3	563.0
85739	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Three bedroom units	v42135561	192.3.4	NaN
85740	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row structures of three units and over	Bachelor units	v42135577	192.2.1	NaN

In [9]:
```
rent_10years_df.isnull().sum()
```

Out[9]:
```
REF_DATE                0
GEO                     0
DGUID                   0
Type of structure       0
Type of unit            0
VECTOR                  0
COORDINATE              0
VALUE               12664
dtype: int64
```

In [10]:
```
rent_10years_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 36624 entries, 85736 to 122359
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   REF_DATE           36624 non-null  int64  
 1   GEO                36624 non-null  object 
 2   DGUID              36624 non-null  object 
 3   Type of structure  36624 non-null  object 
 4   Type of unit       36624 non-null  object 
 5   VECTOR             36624 non-null  object 
 6   COORDINATE         36624 non-null  object 
 7   VALUE              23960 non-null  float64
dtypes: float64(1), int64(1), object(6)
memory usage: 2.5+ MB
```

In [11]:
```
#36624 data, where 12664 of it are NAN, No possible way to recover these data therefore removing NAN in the \
rent_10years_df.dropna(subset=['VALUE'], inplace=True)
rent_10years_df.isnull().sum()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\1190804364.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  rent_10years_df.dropna(subset=['VALUE'], inplace=True)
```

Out[11]:
```
REF_DATE             0
GEO                  0
DGUID                0
Type of structure    0
Type of unit         0
VECTOR               0
COORDINATE           0
VALUE                0
dtype: int64
```

In [12]:
```
rent_10years_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 23960 entries, 85738 to 122359
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   REF_DATE           23960 non-null  int64
 1   GEO                23960 non-null  object
 2   DGUID              23960 non-null  object
 3   Type of structure  23960 non-null  object
 4   Type of unit       23960 non-null  object
 5   VECTOR             23960 non-null  object
 6   COORDINATE         23960 non-null  object
 7   VALUE              23960 non-null  float64
dtypes: float64(1), int64(1), object(6)
memory usage: 1.6+ MB
```

In [162]:
```
# next step is to sperate provience and city from GEO
# rent_10years_df[['city', 'province']] = rent_10years_df['GEO'].str.split(', ', expand=True)
# rent_10years_df.head()
```

In [15]:
```
# next step is to sperate provience and city from GEO
rent_10years_df[['city', 'province']] = rent_10years_df['GEO'].str.split(', ', n=1, expand=True)
rent_10years_df.tail()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\2631396773.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  rent_10years_df[['city', 'province']] = rent_10years_df['GEO'].str.split(', ', n=1, expand=True)
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\2631396773.py:2: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  rent_10years_df[['city', 'province']] = rent_10years_df['GEO'].str.split(', ', n=1, expand=True)
```

Out[15]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province
122355	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of three units and over	Three bedroom units	v3824416	188.1.4	2133.0	Yellowknife	Northwest Territories
122356	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Bachelor units	v3824602	188.4.1	1279.0	Yellowknife	Northwest Territories
122357	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	One bedroom units	v3824790	188.4.2	1563.0	Yellowknife	Northwest Territories
122358	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Two bedroom units	v3824978	188.4.3	1820.0	Yellowknife	Northwest Territories
122359	2022	Yellowknife, Northwest Territories	2011S0504995	Apartment structures of six units and over	Three bedroom units	v3825166	188.4.4	2120.0	Yellowknife	Northwest Territories

In [16]:
```
# first lets get a back up for our data
df=rent_10years_df
df.to_csv('rent_info_all_cities.csv', index=False)

# checking the data to find whats Ottawa is called in the data
rent_10years_df['city'].unique()

#seems like it is called Ottawa-Gatineau, interesting as the rent is lower in Quebec side :) lets isolate the
```

Out[16]:
```
array(['Bay Roberts', 'Corner Brook', 'Gander', 'Grand Falls-Windsor',
       "St. John's", 'Charlottetown', 'Summerside', 'Cape Breton',
       'East Hants', 'Halifax', 'Kentville', 'New Glasgow', 'Queens',
       'Truro', 'Bathurst', 'Campbellton', 'Edmundston', 'Fredericton',
       'Miramichi', 'Moncton', 'Saint John', 'Alma', 'Amos',
       'Baie-Comeau', 'Cowansville', 'Dolbeau-Mistassini',
       'Drummondville', 'Gaspé', 'Ottawa-Gatineau', 'Granby',
       'Hawkesbury', 'Joliette', 'La Tuque', 'Lachute',
       'Les Îles-de-la-Madeleine', 'Matane', 'Mont-Laurier', 'Montmagny',
       'Montréal', 'Prévost', 'Québec', 'Rawdon', 'Rimouski',
       'Rivière-du-Loup', 'Roberval', 'Rouyn-Noranda', 'Saint-Félicien',
       'Saint-Georges', 'Saint-Hyacinthe', 'Saint-Jean-sur-Richelieu',
       'Saint-Lin--Laurentides', 'Sainte-Adèle', 'Sainte-Marie',
       'Sainte- Sophie', 'Saguenay', 'Salaberry-de-Valleyfield',
       'Sept-Îles', 'Shawinigan', 'Sherbrooke', 'Sorel-Tracy',
       'Thetford Mines', 'Trois-Rivières', "Val-d'Or", 'Victoriaville',
       'Barrie', 'Belleville', 'Bracebridge', 'Brantford', 'Brighton',
       'Brock', 'Brockville', 'Centre Wellington', 'Chatham-Kent',
       'Cobourg', 'Collingwood', 'Cornwall', 'Elliot Lake', 'Essex',
       'Gravenhurst', 'Greater Napanee', 'Greater Sudbury', 'Guelph',
       'Haldimand County', 'Hamilton', 'Huntsville', 'Ingersoll',
       'Kawartha Lakes', 'Kenora', 'Kincardine', 'Kings Subdivision',
       'Kingston', 'Kitchener-Cambridge-Waterloo', 'Lambton Shores',
       'Leamington', 'London', 'Meaford', 'Midland', 'Mississippi Mills',
       'Norfolk', 'North Bay', 'North Grenville', 'North Perth',
       'Orillia', 'Oshawa', 'Owen Sound', 'Pembroke', 'Petawawa',
       'Peterborough', 'Port Hope', 'Prince Edward',
       'St. Catharines-Niagara', 'Sarnia', 'Saugeen Shores',
       'Sault Ste. Marie', 'Scugog', 'Stratford', 'Temiskaming Shores',
       'The Nation', 'Thunder Bay', 'Tillsonburg', 'Timmins', 'Toronto',
       'West Grey', 'West Nipissing', 'Windsor', 'Woodstock', 'Brandon',
       'Hanover', 'Portage La Prairie', 'Steinbach', 'Thompson',
       'Winnipeg', 'Estevan', 'Lloydminster', 'Moose Jaw',
       'North Battleford', 'Prince Albert', 'Regina', 'Saskatoon',
       'Swift Current', 'Yorkton', 'Brooks', 'Calgary', 'Camrose',
       'Canmore', 'Cold Lake', 'Edmonton', 'Grande Prairie', 'High River',
       'Lacombe', 'Lethbridge', 'Medicine Hat', 'Okotoks', 'Red Deer',
       'Strathmore', 'Sylvan Lake', 'Wetaskiwin', 'Wood Buffalo',
       'Abbotsford-Mission', 'Campbell River', 'Chilliwack', 'Courtenay',
       'Cranbrook', 'Dawson Creek', 'Duncan', 'Fort St. John', 'Kamloops',
       'Kelowna', 'Nanaimo', 'Parksville', 'Penticton', 'Port Alberni',
       'Powell River', 'Prince George', 'Prince Rupert', 'Quesnel',
       'Salmon Arm', 'Squamish', 'Summerland', 'Terrace', 'Vancouver',
       'Vernon', 'Victoria', 'Williams Lake', 'Yellowknife', 'Yarmouth',
       'West Hants', 'Red Deer County'], dtype=object)
```

In [17]:
```
Type_of_unit=rent_10years_df['Type of unit'].unique()
```

In [18]:
```
Type_of_structure=rent_10years_df['Type of structure'].unique()
```

In [19]:
```
Ottawa_rent_df = rent_10years_df.loc[rent_10years_df['city'] == 'Ottawa-Gatineau']
Ottawa_rent_df['province'].value_counts()
```

Out[19]:
```
Ontario/Quebec                165
Ontario part, Ontario/Quebec  165
Quebec part, Ontario/Quebec   152
Name: province, dtype: int64
```

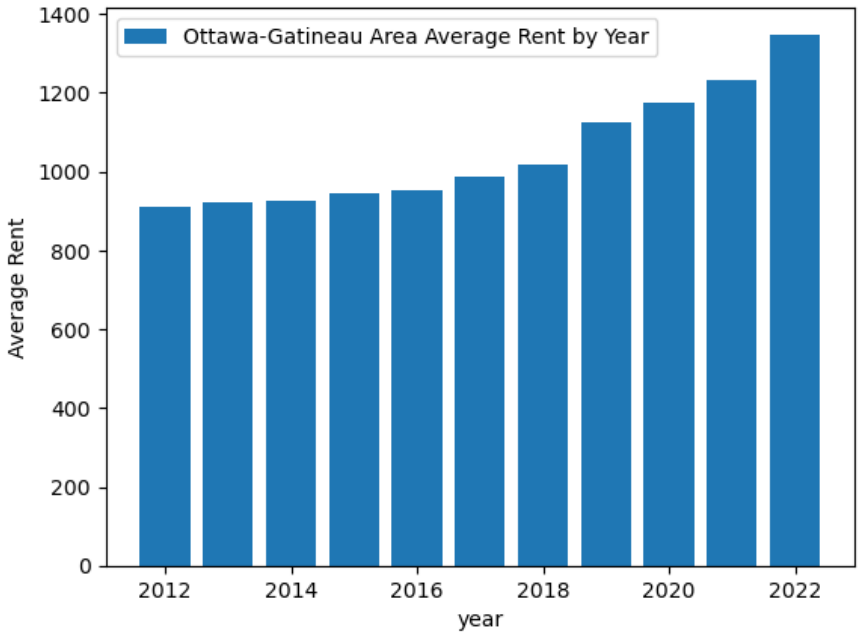
In [21]:
```
#Lets get a back up for Ottawa-Gatineau data
df2=Ottawa_rent_df
df2.to_csv('rent_info_Ottawa.csv', index=False)
```

In [22]:
```
# looks like there is difference between Quebec part and Ontario part
# but some of data is not specific on which part of the river
# lets take a look at those
Ottawa_rent_df.loc[Ottawa_rent_df['province'] == 'Ontario/Quebec'].head()
```

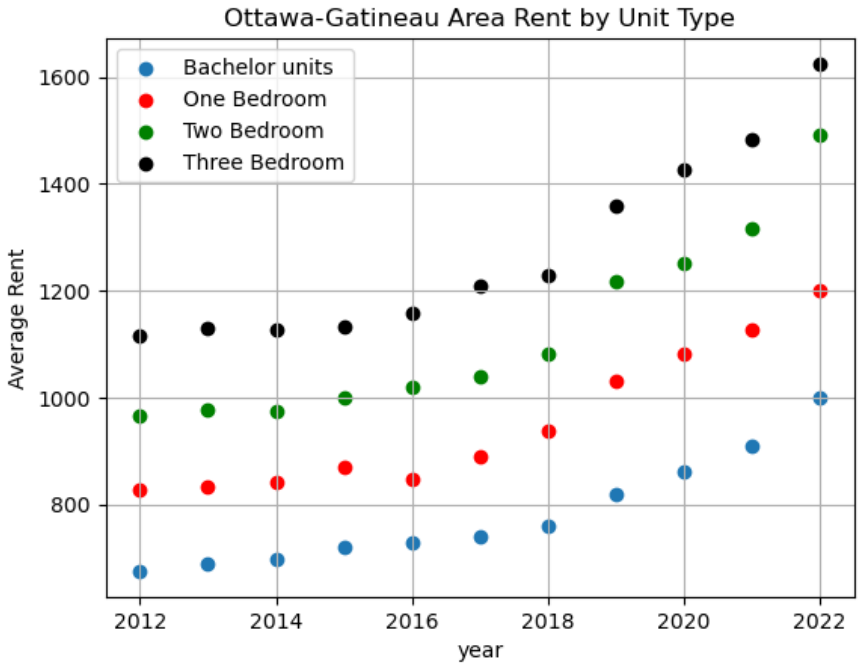
Out[22]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province
87512	2012	Ottawa-Gatineau, Ontario/Quebec	2011S0503505	Row and apartment structures of three units an...	Bachelor units	v3822824	105.3.1	727.0	Ottawa-Gatineau	Ontario/Quebec
87513	2012	Ottawa-Gatineau, Ontario/Quebec	2011S0503505	Row and apartment structures of three units an...	One bedroom units	v3822958	105.3.2	877.0	Ottawa-Gatineau	Ontario/Quebec
87514	2012	Ottawa-Gatineau, Ontario/Quebec	2011S0503505	Row and apartment structures of three units an...	Two bedroom units	v3823092	105.3.3	992.0	Ottawa-Gatineau	Ontario/Quebec
87515	2012	Ottawa-Gatineau, Ontario/Quebec	2011S0503505	Row and apartment structures of three units an...	Three bedroom units	v3823226	105.3.4	1176.0	Ottawa-Gatineau	Ontario/Quebec
87517	2012	Ottawa-Gatineau, Ontario/Quebec	2011S0503505	Row structures of three units and over	One bedroom units	v3823494	105.2.2	888.0	Ottawa-Gatineau	Ontario/Quebec

In [45]:
```
# Let's check if the rent change with year
import matplotlib.pyplot as plt
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df.groupby(['REF_DATE']).mean()
plt.bar(X, Y1['VALUE'], label='Ottawa-Gatineau Area Average Rent by Year')
plt.xlabel('year')
plt.ylabel('Average Rent')
plt.legend()
plt.show()
```



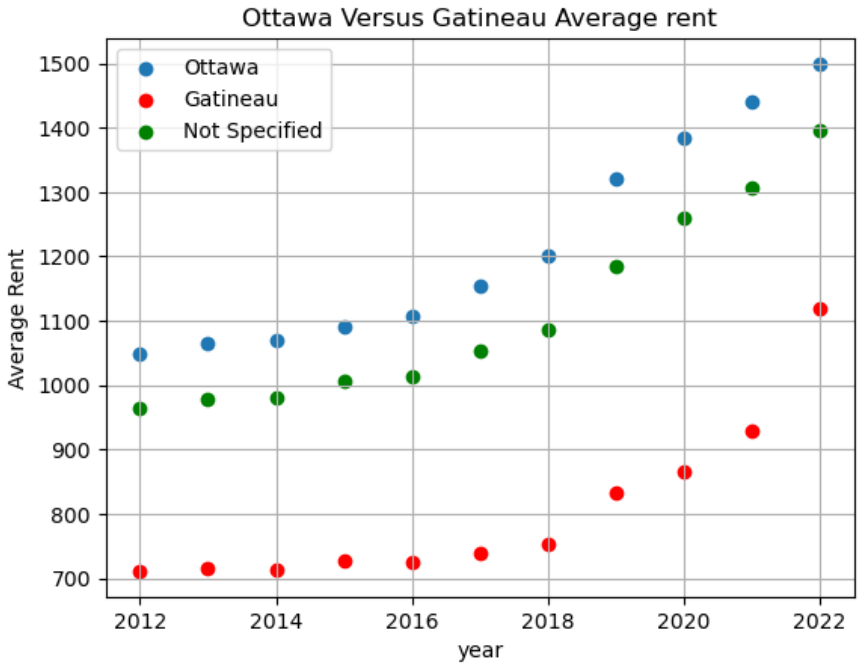
In [53]:
```
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[(Ottawa_rent_df['Type of unit'] == 'Bachelor units')].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[(Ottawa_rent_df['Type of unit'] == 'One bedroom units')].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[(Ottawa_rent_df['Type of unit'] == 'Two bedroom units')].groupby(['REF_DATE']).mean()
Y4=Ottawa_rent_df[(Ottawa_rent_df['Type of unit'] == 'Three bedroom units')].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Bachelor units')
plt.scatter(X, Y2, color='red', label='One Bedroom')
plt.scatter(X, Y3, color='green', label='Two Bedroom')
plt.scatter(X, Y4, color='black', label='Three Bedroom')
plt.xlabel('year')
plt.ylabel('Average Rent')
plt.legend()
plt.grid(True)
plt.title('Ottawa-Gatineau Area Rent by Unit Type')
plt.show()
```



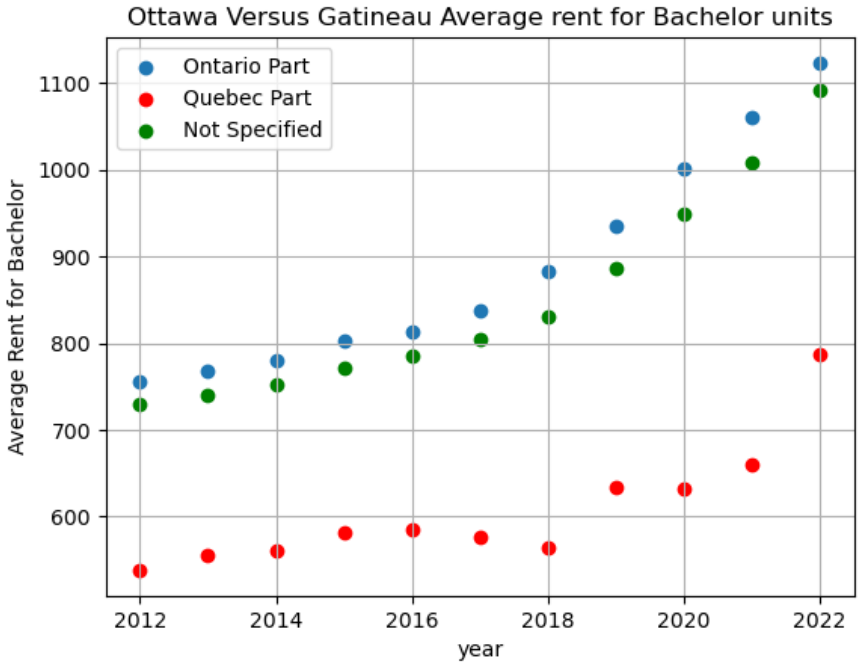

```
# Each year we have data that is unknown on which side of the river and it is one-third of our data !
# I can think of three options:
# Option 1, ignore the data and lose 1/3 of the data! (not good)
# Option 2, Looking at coordinate and prices (visually), it looks all the data goes to Ontario part
# Option 3, keep them separate and anlysis them seprately. (This is what I decided to go with)
# Option 4, assign the unspesified data to one group (not good as well as we are reducing Ottawa rent by it)

# for option 4, there are some assumption that we need to make!
# assumption 1: Rent is lower in Gatineau
# assumption 2: Year and type of unit have an effect on the price
# assumption 3: Type of structure does not have an effect on the price
# Considering these assumptions our plan of action can be :
#1- Calculate the average rent for each type of unit in each year for our known data (Quebec and Ontario side)
#2- Compare the rent that is given in our unknown data to the average that is calculated before (depending on
unit type and year)
#3- Write an if statement to assign Quebec or Ontario to our unknown data
```

In [54]:

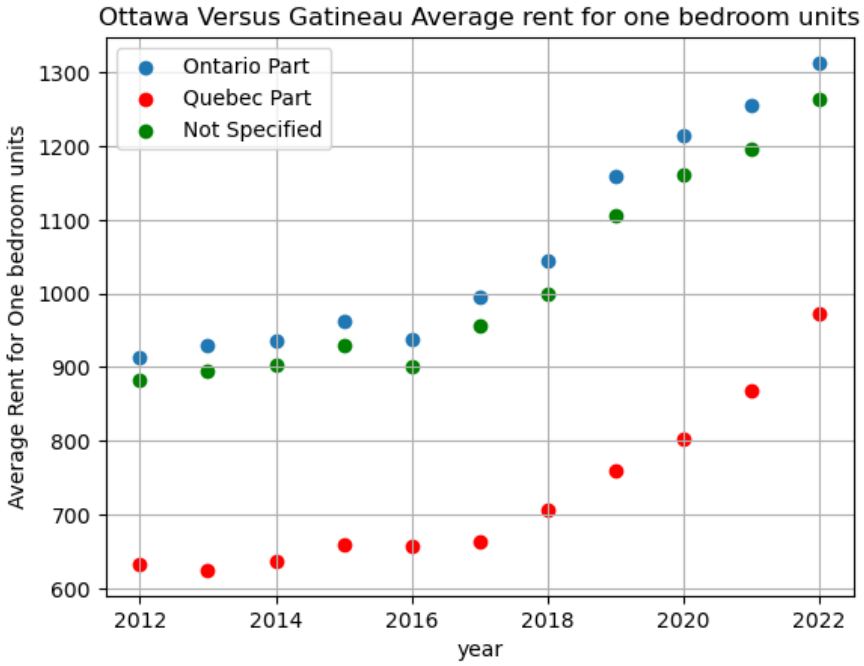
```
# Let's check if the rent in Quebec part is lower than Ontario part
import matplotlib.pyplot as plt
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[Ottawa_rent_df['province']=='Ontario part, Ontario/Quebec'].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[Ottawa_rent_df['province']=='Quebec part, Ontario/Quebec'].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[Ottawa_rent_df['province']=='Ontario/Quebec'].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Ottawa')
plt.scatter(X, Y2, color='red', label='Gatineau')
plt.scatter(X, Y3, color='green', label='Not Specified')
plt.xlabel('year')
plt.ylabel('Average Rent')
plt.title('Ottawa Versus Gatineau Average rent')
plt.grid(True)
plt.legend()
plt.show()
```



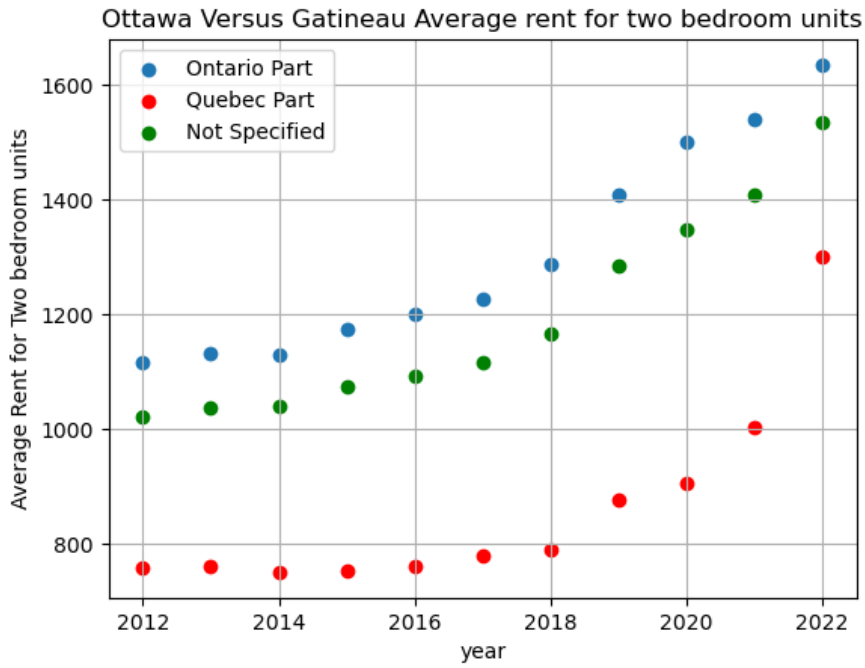
In [55]:
```
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Bachelor units')].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Quebec part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Bachelor units')].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Bachelor units')].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Ontario Part')
plt.scatter(X, Y2, color='red', label='Quebec Part')
plt.scatter(X, Y3, color='green', label='Not Specified')
plt.xlabel('year')
plt.ylabel('Average Rent for Bachelor')
plt.legend()
plt.grid(True)
plt.title('Ottawa Versus Gatineau Average rent for Bachelor units')
plt.show()
```



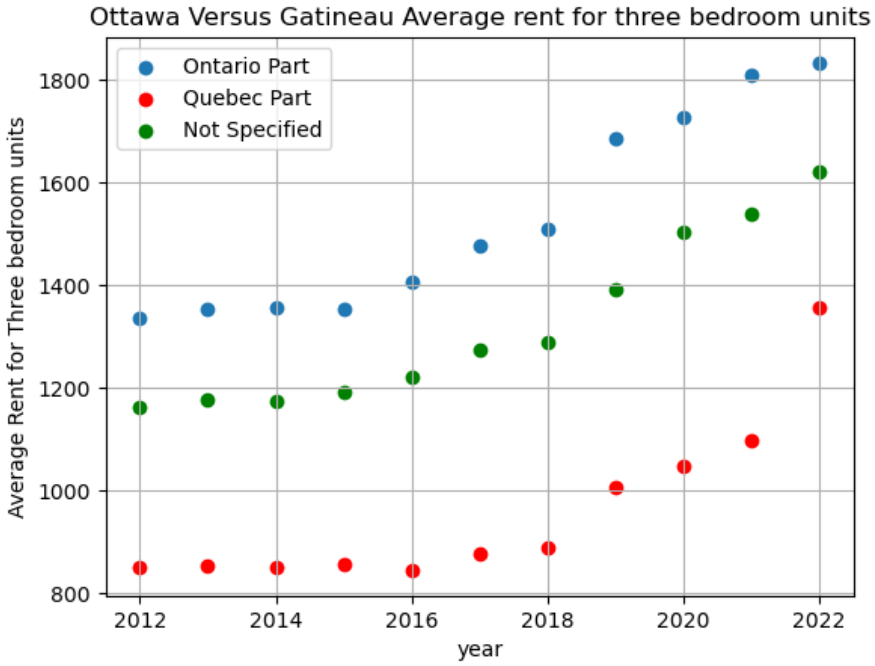
In [57]:
```
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'One bedroom units')].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Quebec part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'One bedroom units')].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'One bedroom units')].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Ontario Part')
plt.scatter(X, Y2, color='red', label='Quebec Part')
plt.scatter(X, Y3, color='green', label='Not Specified')
plt.xlabel('year')
plt.ylabel('Average Rent for One bedroom units')
plt.title('Ottawa Versus Gatineau Average rent for one bedroom units')
plt.grid(True)
plt.legend()
plt.show()
```



In [58]:
```
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Two bedroom units')].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Quebec part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Two bedroom units')].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Two bedroom units')].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Ontario Part')
plt.scatter(X, Y2, color='red', label='Quebec Part')
plt.scatter(X, Y3, color='green', label='Not Specified')
plt.xlabel('year')
plt.ylabel('Average Rent for Two bedroom units')
plt.title('Ottawa Versus Gatineau Average rent for two bedroom units')
plt.grid(True)
plt.legend()
plt.show()
```



In [59]:
```
X=Ottawa_rent_df['REF_DATE'].unique()
Y1=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Three bedroom units')].groupby(['REF_DATE']).mean()
Y2=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Quebec part, Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Three bedroom units')].groupby(['REF_DATE']).mean()
Y3=Ottawa_rent_df[(Ottawa_rent_df['province'] == 'Ontario/Quebec')
                & (Ottawa_rent_df['Type of unit'] == 'Three bedroom units')].groupby(['REF_DATE']).mean()

plt.scatter(X, Y1, label='Ontario Part')
plt.scatter(X, Y2, color='red', label='Quebec Part')
plt.scatter(X, Y3, color='green', label='Not Specified')
plt.xlabel('year')
plt.ylabel('Average Rent for Three bedroom units')
plt.title('Ottawa Versus Gatineau Average rent for three bedroom units')
plt.grid(True)
plt.legend()
plt.show()
```



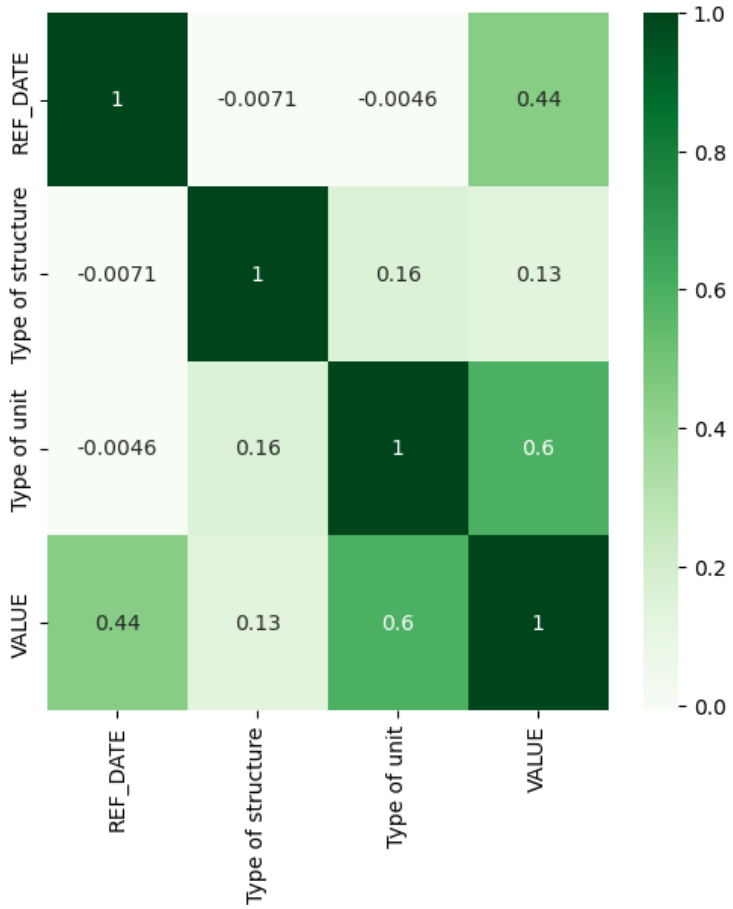
In [63]:
```
# changing the Unit type and structure to numbers, so we can undrestand rent dependency
Ottawa_rent_df2=Ottawa_rent_df
import seaborn as sns
Ottawa_rent_df2['Type of unit'] = Ottawa_rent_df2['Type of unit'].replace({'Bachelor units': 0, 'One bedroom
                                                                           'Two bedroom units': 2,'Three bedro

Ottawa_rent_df2['Type of structure'] = Ottawa_rent_df2['Type of structure'].replace({'Row and apartment stru
                                                                                    'Apartment structures of
                                                                                    'Apartment structures of
                                                                                    'Row structures of three
heatmap_df=Ottawa_rent_df2[['REF_DATE','Type of structure','Type of unit','VALUE']]


heatmap_df.head()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\289126956.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  Ottawa_rent_df2['Type of unit'] = Ottawa_rent_df2['Type of unit'].replace({'Bachelor units': 0, 'One bedr
oom units': 1,
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\289126956.py:7: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  Ottawa_rent_df2['Type of structure'] = Ottawa_rent_df2['Type of structure'].replace({'Row and apartment s
tructures of three units and over': 0,
```

Out[63]:

	REF_DATE	Type of structure	Type of unit	VALUE
86264	2012	0	0	528.0
86265	2012	0	1	628.0
86266	2012	0	2	744.0
86267	2012	0	3	835.0
86270	2012	3	2	795.0

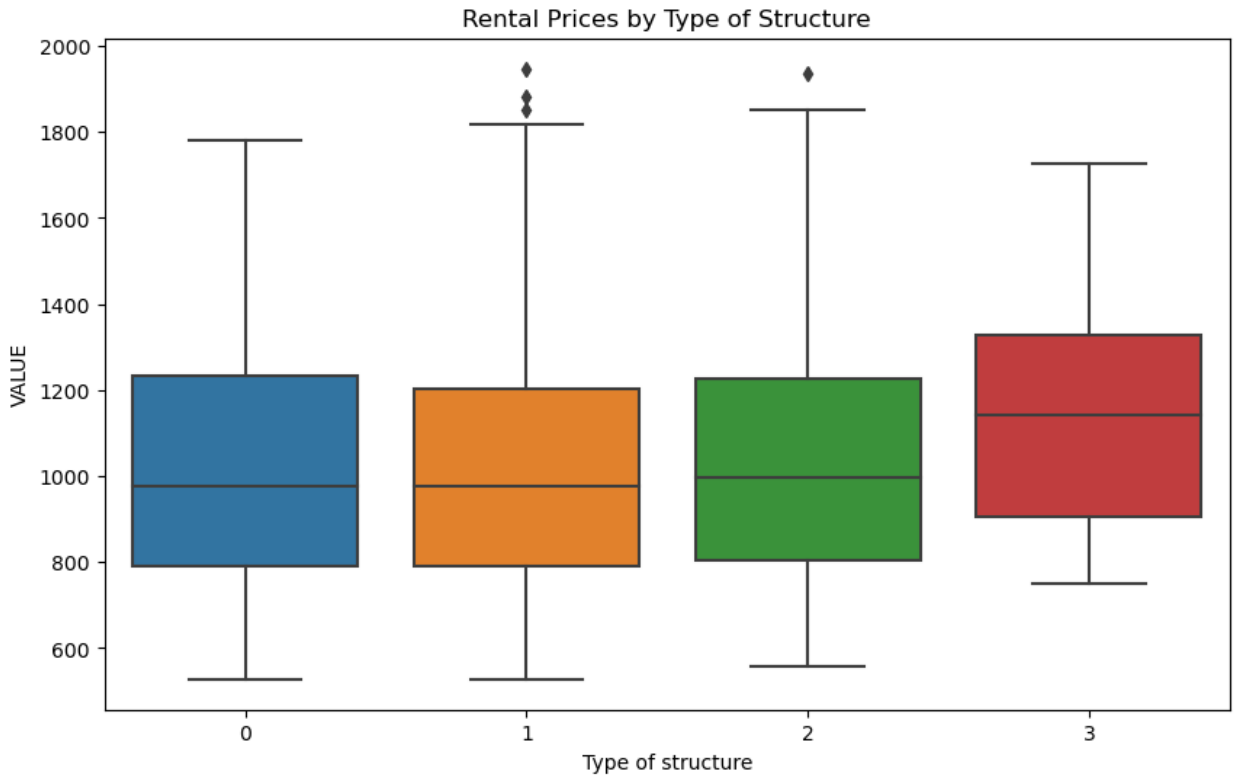
In [64]:
```
# Heat map of infuence of the data on eachother
f, ax = plt.subplots(1, 1, figsize=(6, 6))
ax = sns.heatmap(heatmap_df.corr(), annot=True, cmap='Greens')
```



In [66]:
```
heatmap_df.corr()['VALUE'].sort_values()
```

Out[66]:
```
Type of structure    0.128759
REF_DATE             0.442930
Type of unit         0.598182
VALUE                1.000000
Name: VALUE, dtype: float64
```

In [68]:
```
# Create a boxplot to visualize rental prices by type of structure
plt.figure(figsize=(10, 6))
sns.boxplot(x='Type of structure', y='VALUE', data=Ottawa_rent_df)
plt.title('Rental Prices by Type of Structure')
plt.show()
```



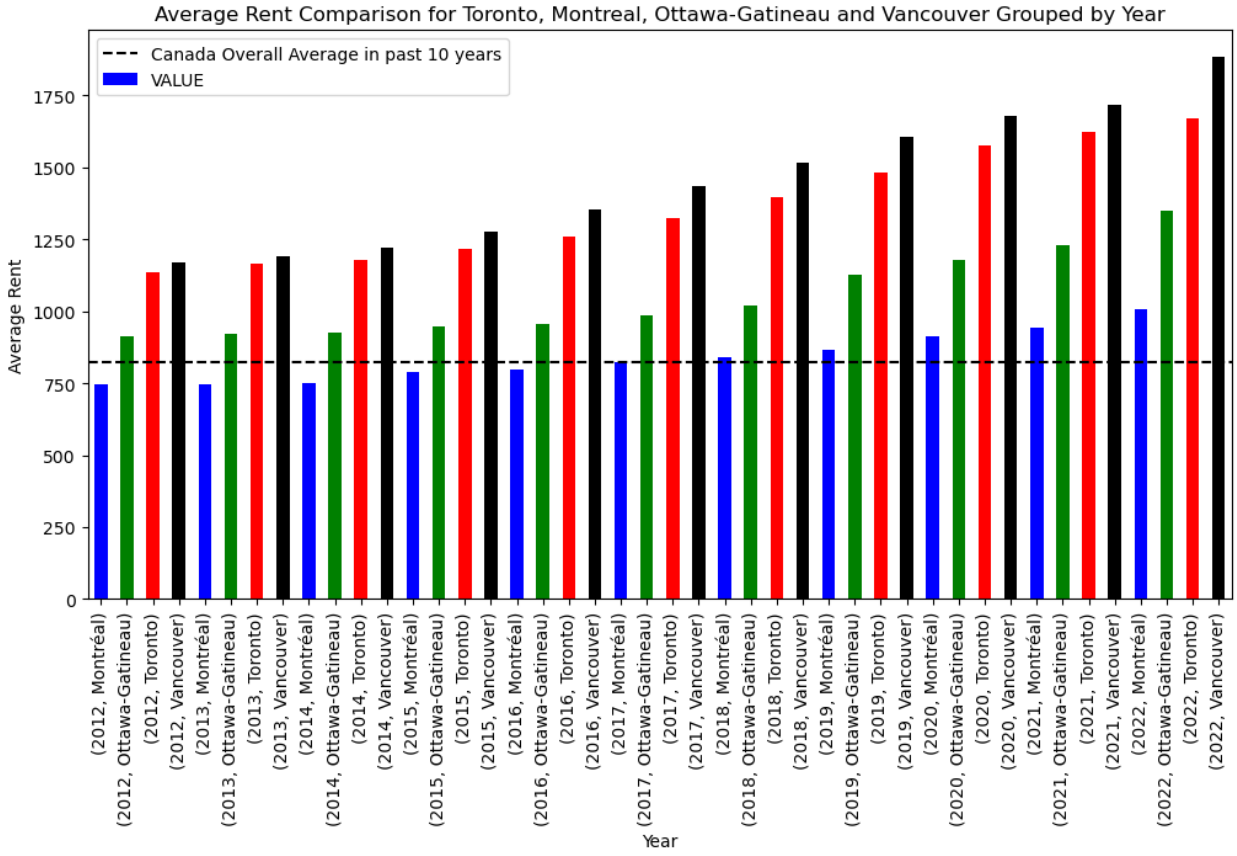
In [69]:
```
rent_10years_df.head()
```

Out[69]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province
85738	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Row and apartment structures of three units an...	Two bedroom units	v42135545	192.3.3	563.0	Bay Roberts	Newfoundland and Labrador
85746	2012	Bay Roberts, Newfoundland and Labrador	2011S0504005	Apartment structures of three units and over	Two bedroom units	v42135673	192.1.3	563.0	Bay Roberts	Newfoundland and Labrador
85752	2012	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Bachelor units	v3822895	2.3.1	439.0	Corner Brook	Newfoundland and Labrador
85753	2012	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	One bedroom units	v3823029	2.3.2	521.0	Corner Brook	Newfoundland and Labrador
85754	2012	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Two bedroom units	v3823163	2.3.3	610.0	Corner Brook	Newfoundland and Labrador

In [159]:

```
#
overall_average = rent_10years_df['VALUE'].mean()
selected_cities = ['Toronto', 'Montréal', 'Vancouver','Ottawa-Gatineau']
city_comparison = rent_10years_df[rent_10years_df['city'].isin(selected_cities)]
# Group by city and calculate average rent for each
grouped_data  = city_comparison.groupby(['REF_DATE','city'])['VALUE'].mean()

grouped_data
# # Plot comparison

plt.figure(figsize=(12, 6))
grouped_data.plot(kind='bar', color=['blue', 'green', 'red','black'])
plt.axhline(y=overall_average, color='black', linestyle='--', label='Canada Overall Average in past 10 years
plt.title('Average Rent Comparison for Toronto, Montreal, Ottawa-Gatineau and Vancouver Grouped by Year')
plt.xlabel('Year')
plt.ylabel('Average Rent')
plt.legend()
plt.show()
```



In [72]:
```
#CPI for Ottawa
Ottawa_rent_df
# Select a base year
base_year = 2012

# Calculate the average rent for each city in the base year
base_year_data = Ottawa_rent_df[Ottawa_rent_df['REF_DATE'] == base_year]
base_year_avg_rent = base_year_data['VALUE'].mean()

# Merge the base year average rent with the original DataFrame
Ottawa_rent_df['MEAN_BASE_YEAR']=base_year_avg_rent

# Calculate CPI
Ottawa_rent_df['CPI'] = (Ottawa_rent_df['VALUE'] / Ottawa_rent_df['MEAN_BASE_YEAR']) * 100

# Display the resulting DataFrame
Ottawa_rent_df.head()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\603455586.py:11: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  Ottawa_rent_df['MEAN_BASE_YEAR']=base_year_avg_rent
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\603455586.py:14: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  Ottawa_rent_df['CPI'] = (Ottawa_rent_df['VALUE'] / Ottawa_rent_df['MEAN_BASE_YEAR']) * 100
```

Out[72]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province	MEAN
86264	2012	Ottawa-Gatineau, Quebec part, Ontario/Quebec	2011S050524505	0	0	v3822825	37.3.1	528.0	Ottawa-Gatineau	Quebec part, Ontario/Quebec	
86265	2012	Ottawa-Gatineau, Quebec part, Ontario/Quebec	2011S050524505	0	1	v3822959	37.3.2	628.0	Ottawa-Gatineau	Quebec part, Ontario/Quebec	
86266	2012	Ottawa-Gatineau, Quebec part, Ontario/Quebec	2011S050524505	0	2	v3823093	37.3.3	744.0	Ottawa-Gatineau	Quebec part, Ontario/Quebec	
86267	2012	Ottawa-Gatineau, Quebec part, Ontario/Quebec	2011S050524505	0	3	v3823227	37.3.4	835.0	Ottawa-Gatineau	Quebec part, Ontario/Quebec	
86270	2012	Ottawa-Gatineau, Quebec part, Ontario/Quebec	2011S050524505	3	2	v3823629	37.2.3	795.0	Ottawa-Gatineau	Quebec part, Ontario/Quebec	

In [116]:
```
file_path=("34100133-eng/18100005.csv")
CPI_df=pd.read_csv(file_path)
CPI_df.head()
```

Out[116]:

	REF_DATE	GEO	DGUID	Products and product groups	UOM	UOM_ID	SCALAR_FACTOR	SCALAR_ID	VECTOR	COORDINATE	VA
0	1914	Canada	2016A000011124	All-items	2002=100	17	units	0	v41693271	2.200	
1	1914	Canada	2016A000011124	All-items (1992=100)	1992=100	7	units	0	v41713433	2.309	
2	1914	Canada	2016A000011124	Goods and services	2002=100	17	units	0	v41693519	2.273	
3	1915	Canada	2016A000011124	All-items	2002=100	17	units	0	v41693271	2.200	
4	1915	Canada	2016A000011124	All-items (1992=100)	1992=100	7	units	0	v41713433	2.309	

In [117]:
```
# no useful data on these columns, therefor dropping them
columns_to_drop = ['UOM','UOM_ID', 'SCALAR_FACTOR', 'SCALAR_ID', 'STATUS','SYMBOL', 'TERMINATED','DECIMALS']
CPI_df.drop(columns=columns_to_drop, inplace=True)
CPI_df.head()
```

Out[117]:

	REF_DATE	GEO	DGUID	Products and product groups	VECTOR	COORDINATE	VALUE
0	1914	Canada	2016A000011124	All-items	v41693271	2.200	6.0
1	1914	Canada	2016A000011124	All-items (1992=100)	v41713433	2.309	7.2
2	1914	Canada	2016A000011124	Goods and services	v41693519	2.273	6.0
3	1915	Canada	2016A000011124	All-items	v41693271	2.200	6.1
4	1915	Canada	2016A000011124	All-items (1992=100)	v41713433	2.309	7.3

In [118]:
```
CPI_df = CPI_df[CPI_df['REF_DATE'] >= 2000]
CPI_df.head()
```

Out[118]:

	REF_DATE	GEO	DGUID	Products and product groups	VECTOR	COORDINATE	VALUE
41286	2000	Canada	2016A000011124	All-items	v41693271	2.2	95.4
41287	2000	Canada	2016A000011124	Food	v41693272	2.3	93.3
41288	2000	Canada	2016A000011124	Food purchased from stores	v41693273	2.4	93.0
41289	2000	Canada	2016A000011124	Meat	v41693274	2.5	90.8
41290	2000	Canada	2016A000011124	Fresh or frozen meat (excluding poultry)	v41693275	2.6	86.7

In [119]:
```
CPI_df['Products and product groups'].unique()
```

```
       'Breakfast cereal and other cereal products (excluding baby food)',
       'Pasta products', 'Flour and flour-based mixes',
       'Fruit, fruit preparations and nuts', 'Fresh fruit', 'Apples',
       'Oranges', 'Bananas', 'Other fresh fruit',
       'Preserved fruit and fruit preparations', 'Fruit juices',
       'Other preserved fruit and fruit preparations', 'Nuts and seeds',
       'Vegetables and vegetable preparations', 'Fresh vegetables',
       'Potatoes', 'Tomatoes', 'Lettuce', 'Other fresh vegetables',
       'Preserved vegetables and vegetable preparations',
       'Frozen and dried vegetables',
       'Canned vegetables and other vegetable preparations',
       'Other food products and non-alcoholic beverages',
       'Sugar and confectionery', 'Sugar and syrup', 'Confectionery',
       'Edible fats and oils', 'Margarine', 'Other edible fats and oils',
       'Coffee and tea', 'Coffee', 'Tea',
       'Condiments, spices and vinegars', 'Other food preparations',
       'Soup', 'Baby foods', 'Frozen food preparations',
       'All other food preparations', 'Non-alcoholic beverages',
       'Food purchased from restaurants',
       'Food purchased from table-service restaurants'
```

In [120]:
```
CPI_df[CPI_df['Products and product groups']=='Rented accommodation']
```

Out[120]:

	REF_DATE	GEO	DGUID	Products and product groups	VECTOR	COORDINATE	VALUE
41364	2000	Canada	2016A000011124	Rented accommodation	v41693349	2.8	96.5
41596	2000	Newfoundland and Labrador	2016A000210	Rented accommodation	v41693576	3.8	97.3
41702	2000	St. John's, Newfoundland and Labrador	2011S0503001	Rented accommodation	v41695146	4.8	96.7
41740	2000	Prince Edward Island	2016A000211	Rented accommodation	v41693711	5.8	97.6
41845	2000	Charlottetown and Summerside, Prince Edward Is...	NaN	Rented accommodation	v41695152	6.8	97.8
...	...	...	...	...	...	...	...
87826	2022	Edmonton, Alberta	2011S0503835	Rented accommodation	v41695218	24.8	148.9
87831	2022	Calgary, Alberta	2011S0503825	Rented accommodation	v41695224	25.8	133.1
87868	2022	British Columbia	2016A000259	Rented accommodation	v41694794	26.8	136.1
87971	2022	Vancouver, British Columbia	2011S0503933	Rented accommodation	v41695230	27.8	140.4
87976	2022	Victoria, British Columbia	2011S0503935	Rented accommodation	v41695236	28.8	137.0

621 rows × 7 columns

In [121]:
```
CPI_df['GEO'].unique()
```

Out[121]:
```
array(['Canada', 'Newfoundland and Labrador',
       "St. John's, Newfoundland and Labrador", 'Prince Edward Island',
       'Charlottetown and Summerside, Prince Edward Island',
       'Nova Scotia', 'Halifax, Nova Scotia', 'New Brunswick',
       'Saint John, New Brunswick', 'Quebec', 'Québec, Quebec',
       'Montréal, Quebec', 'Ontario',
       'Ottawa-Gatineau, Ontario part, Ontario/Quebec',
       'Toronto, Ontario', 'Thunder Bay, Ontario', 'Manitoba',
       'Winnipeg, Manitoba', 'Saskatchewan', 'Regina, Saskatchewan',
       'Saskatoon, Saskatchewan', 'Alberta', 'Edmonton, Alberta',
       'Calgary, Alberta', 'British Columbia',
       'Vancouver, British Columbia', 'Victoria, British Columbia',
       'Whitehorse, Yukon', 'Yellowknife, Northwest Territories',
       'Iqaluit, Nunavut'], dtype=object)
```

In [122]:
```
# next step is to sperate provience and city from GEO
CPI_df[['city', 'province']] = CPI_df['GEO'].str.split(', ', n=1, expand=True)
CPI_df.tail()
```

Out[122]:

	REF_DATE	GEO	DGUID	Products and product groups	VECTOR	COORDINATE	VALUE	city	province
88223	2022	Yellowknife, Northwest Territories	2011A00056106023	Durable goods	v41695129	30.275	111.6	Yellowknife	Northwest Territories
88224	2022	Yellowknife, Northwest Territories	2011A00056106023	Semi-durable goods	v41695130	30.276	109.2	Yellowknife	Northwest Territories
88225	2022	Yellowknife, Northwest Territories	2011A00056106023	Non-durable goods	v41695131	30.277	179.4	Yellowknife	Northwest Territories
88226	2022	Yellowknife, Northwest Territories	2011A00056106023	Services	v41695132	30.282	153.3	Yellowknife	Northwest Territories
88227	2022	Iqaluit, Nunavut	2011A00056204003	All-items	v41713462	31.200	138.4	Iqaluit	Nunavut

In [125]:
```
CPI_df[(CPI_df['Products and product groups']=='Shelter')
       & (CPI_df['city']=='Canada')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada
```

Out[125]:

	REF_DATE	GEO	DGUID	Products and product groups	VECTOR	COORDINATE	VALUE	city	province
41363	2000	Canada	2016A000011124	Shelter	v41693348	2.79	95.6	Canada	None
43373	2001	Canada	2016A000011124	Shelter	v41693348	2.79	99.1	Canada	None
45383	2002	Canada	2016A000011124	Shelter	v41693348	2.79	100.0	Canada	None
47394	2003	Canada	2016A000011124	Shelter	v41693348	2.79	103.2	Canada	None
49432	2004	Canada	2016A000011124	Shelter	v41693348	2.79	105.8	Canada	None
51470	2005	Canada	2016A000011124	Shelter	v41693348	2.79	109.2	Canada	None
53508	2006	Canada	2016A000011124	Shelter	v41693348	2.79	113.1	Canada	None
55546	2007	Canada	2016A000011124	Shelter	v41693348	2.79	116.9	Canada	None
57584	2008	Canada	2016A000011124	Shelter	v41693348	2.79	122.0	Canada	None
59623	2009	Canada	2016A000011124	Shelter	v41693348	2.79	121.6	Canada	None
61662	2010	Canada	2016A000011124	Shelter	v41693348	2.79	123.3	Canada	None
63701	2011	Canada	2016A000011124	Shelter	v41693348	2.79	125.6	Canada	None
65740	2012	Canada	2016A000011124	Shelter	v41693348	2.79	127.1	Canada	None
67812	2013	Canada	2016A000011124	Shelter	v41693348	2.79	128.7	Canada	None
69902	2014	Canada	2016A000011124	Shelter	v41693348	2.79	132.2	Canada	None
71992	2015	Canada	2016A000011124	Shelter	v41693348	2.79	133.7	Canada	None
74082	2016	Canada	2016A000011124	Shelter	v41693348	2.79	135.8	Canada	None
76173	2017	Canada	2016A000011124	Shelter	v41693348	2.79	138.1	Canada	None
78260	2018	Canada	2016A000011124	Shelter	v41693348	2.79	140.9	Canada	None
80347	2019	Canada	2016A000011124	Shelter	v41693348	2.79	144.5	Canada	None
82369	2020	Canada	2016A000011124	Shelter	v41693348	2.79	147.0	Canada	None
84386	2021	Canada	2016A000011124	Shelter	v41693348	2.79	152.7	Canada	None
86359	2022	Canada	2016A000011124	Shelter	v41693348	2.79	163.3	Canada	None

In [129]:
```
# Lets create a CPI with base year of 2002 for our rent data
rent_after2002_df = rent_df[rent_df['REF_DATE'] >= 2002]
rent_after2002_df.dropna(subset=['VALUE'], inplace=True)
rent_after2002_df[['city', 'province']] = rent_after2002_df['GEO'].str.split(', ', n=1, expand=True)
rent_after2002_df.head()
```

```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3710110094.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  rent_after2002_df.dropna(subset=['VALUE'], inplace=True)
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3710110094.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  rent_after2002_df[['city', 'province']] = rent_after2002_df['GEO'].str.split(', ', n=1, expand=True)
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3710110094.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  rent_after2002_df[['city', 'province']] = rent_after2002_df['GEO'].str.split(', ', n=1, expand=True)
```

Out[129]:

	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province
51865	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	One bedroom units	v3823029	2.3.2	383.0	Corner Brook	Newfoundland and Labrador
51866	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Two bedroom units	v3823163	2.3.3	436.0	Corner Brook	Newfoundland and Labrador
51867	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Three bedroom units	v3823297	2.3.4	514.0	Corner Brook	Newfoundland and Labrador
51873	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Apartment structures of three units and over	One bedroom units	v3824101	2.1.2	384.0	Corner Brook	Newfoundland and Labrador
51874	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Apartment structures of three units and over	Two bedroom units	v3824235	2.1.3	436.0	Corner Brook	Newfoundland and Labrador

In [139]:
```
# Select a base year
base_year = 2002

# Calculate the average rent for each city in the base year
base_year_data = rent_after2002_df[rent_after2002_df['REF_DATE'] == base_year]
base_year_avg_rent = base_year_data['VALUE'].mean()

# Merge the base year average rent with the original DataFrame
rent_after2002_df['MEAN_BASE_YEAR']=base_year_avg_rent

# Calculate CPI
rent_after2002_df['CPI'] = (rent_after2002_df['VALUE'] / rent_after2002_df['MEAN_BASE_YEAR']) * 100

# Display the resulting DataFrame
rent_after2002_df.head()
```

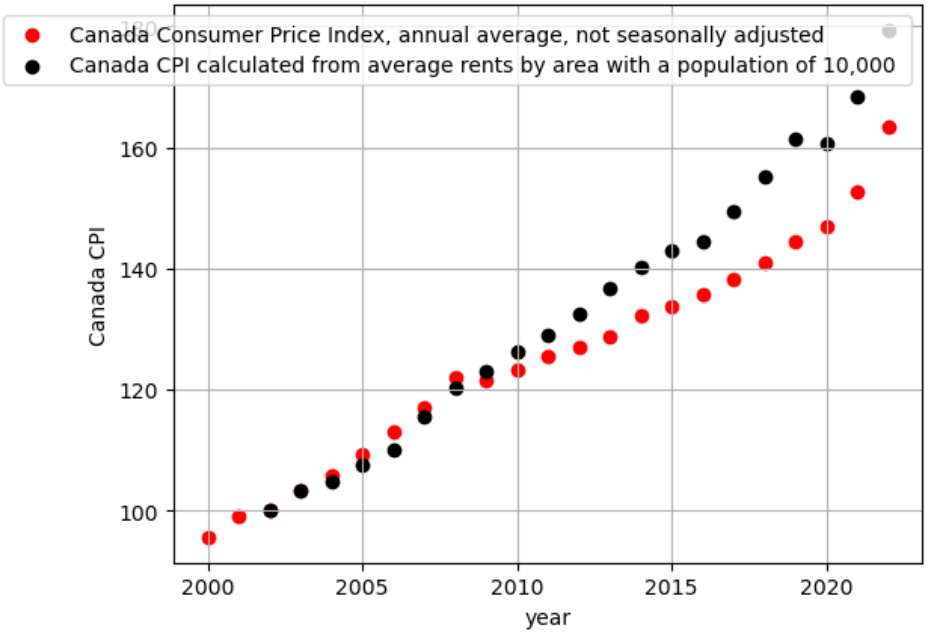
```
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3963579102.py:9: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  rent_after2002_df['MEAN_BASE_YEAR']=base_year_avg_rent
C:\Users\rozap\AppData\Local\Temp\ipykernel_20124\3963579102.py:12: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html
#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#retu
rning-a-view-versus-a-copy)
  rent_after2002_df['CPI'] = (rent_after2002_df['VALUE'] / rent_after2002_df['MEAN_BASE_YEAR']) * 100
```

Out[139]:

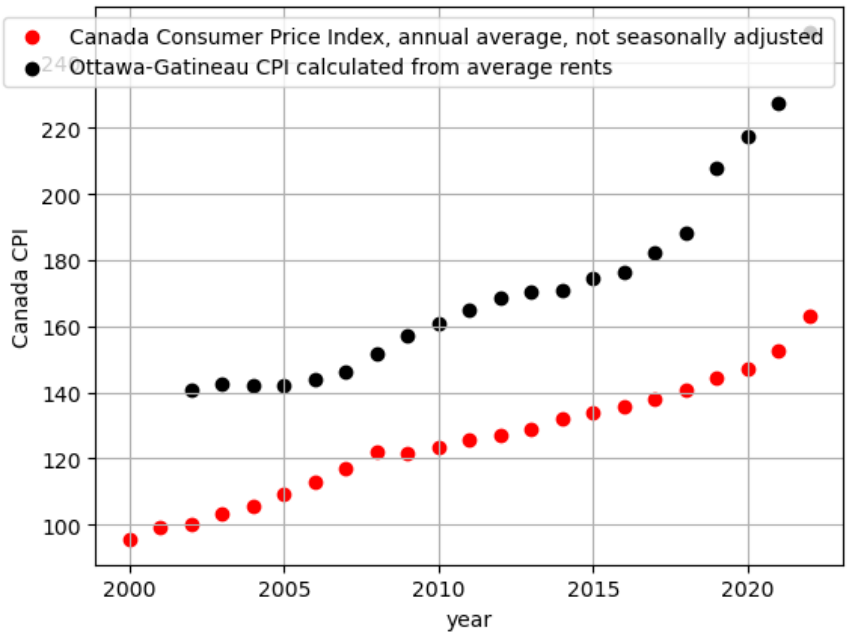
	REF_DATE	GEO	DGUID	Type of structure	Type of unit	VECTOR	COORDINATE	VALUE	city	province	MEAN_BA
51865	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	One bedroom units	v3823029	2.3.2	383.0	Corner Brook	Newfoundland and Labrador	
51866	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Two bedroom units	v3823163	2.3.3	436.0	Corner Brook	Newfoundland and Labrador	
51867	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Row and apartment structures of three units an...	Three bedroom units	v3823297	2.3.4	514.0	Corner Brook	Newfoundland and Labrador	
51873	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Apartment structures of three units and over	One bedroom units	v3824101	2.1.2	384.0	Corner Brook	Newfoundland and Labrador	
51874	2002	Corner Brook, Newfoundland and Labrador	2011S0504015	Apartment structures of three units and over	Two bedroom units	v3824235	2.1.3	436.0	Corner Brook	Newfoundland and Labrador	

In [161]:

```
# now lets compare:
A=CPI_df[(CPI_df['Products and product groups']=='Shelter')
       & (CPI_df['city']=='Canada')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada

X=A['REF_DATE']
Y=A['VALUE']
plt.scatter(X, Y , color='red', label='Canada Consumer Price Index, annual average, not seasonally adjusted'
B=rent_after2002_df.groupby(['REF_DATE']).mean()
Y2=B['CPI']
X2=rent_after2002_df['REF_DATE'].unique()
plt.scatter(X2, Y2, color='black', label= 'Canada CPI calculated from average rents by area with a populatio
plt.xlabel('year')
plt.ylabel('Canada CPI')
plt.grid(True)
plt.legend()
plt.show()
```

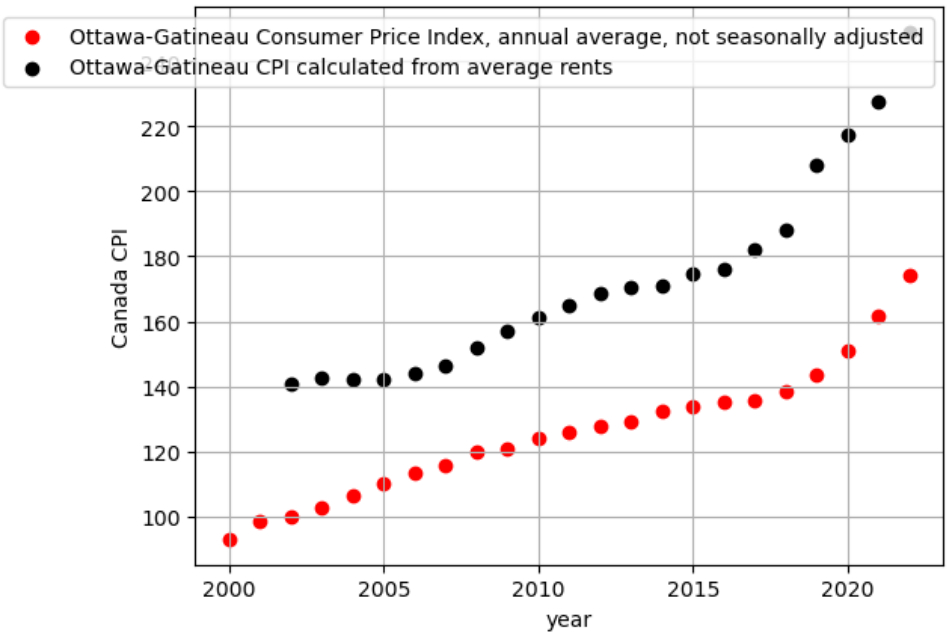


In [150]:
```
# now lets compare Ottawa-Gatineau area to Canada:
A=CPI_df[(CPI_df['Products and product groups']=='Shelter')
       & (CPI_df['city']=='Canada')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada

X=A['REF_DATE']
Y=A['VALUE']
plt.scatter(X, Y , color='red', label='Canada Consumer Price Index, annual average, not seasonally adjusted'
B=rent_after2002_df[rent_after2002_df['city']=='Ottawa-Gatineau'].groupby(['REF_DATE']).mean()
Y2=B['CPI']
X2=rent_after2002_df['REF_DATE'].unique()
plt.scatter(X2, Y2, color='black', label= 'Ottawa-Gatineau CPI calculated from average rents')
plt.xlabel('year')
plt.ylabel('Canada CPI')
plt.grid(True)
plt.legend()
plt.show()
```



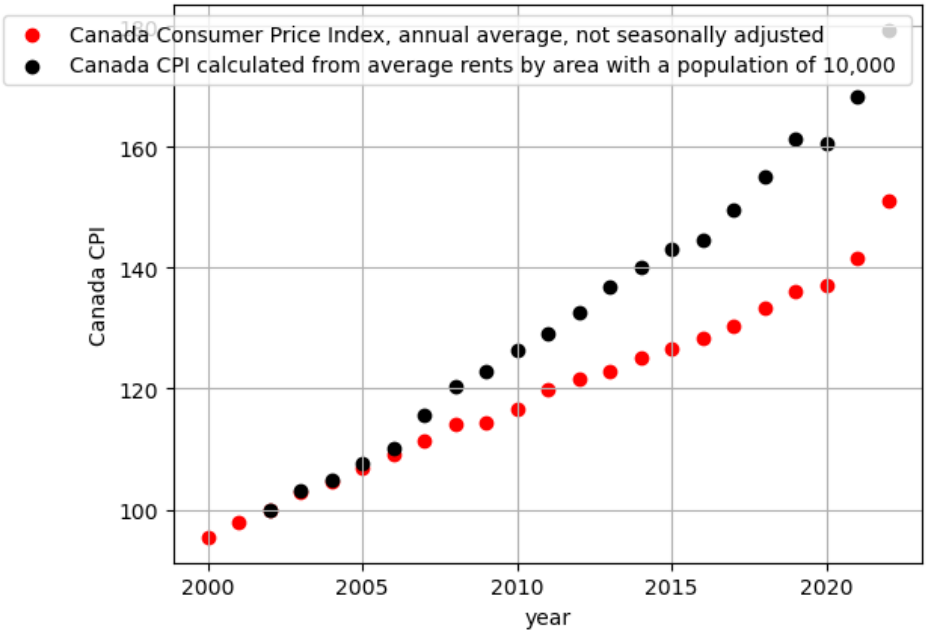
In [155]:

```
# now lets compare Ottawa-Gatineau area to Canada:
A=CPI_df[(CPI_df['Products and product groups']=='Shelter')
       & (CPI_df['city']=='Ottawa-Gatineau')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada

X=A['REF_DATE']
Y=A['VALUE']
plt.scatter(X, Y , color='red', label='Ottawa-Gatineau Consumer Price Index, annual average, not seasonally a
B=rent_after2002_df[rent_after2002_df['city']=='Ottawa-Gatineau'].groupby(['REF_DATE']).mean()
Y2=B['CPI']
X2=rent_after2002_df['REF_DATE'].unique()
plt.scatter(X2, Y2, color='black', label= 'Ottawa-Gatineau CPI calculated from average rents')
plt.xlabel('year')
plt.ylabel('Canada CPI')
plt.grid(True)
plt.legend()
plt.show()
```



In [156]:

```
# now lets compare:
A=CPI_df[(CPI_df['Products and product groups']=='All-items')
       & (CPI_df['city']=='Canada')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada

X=A['REF_DATE']
Y=A['VALUE']
plt.scatter(X, Y , color='red', label='Canada Consumer Price Index, annual average, not seasonally adjusted'
B=rent_after2002_df.groupby(['REF_DATE']).mean()
Y2=B['CPI']
X2=rent_after2002_df['REF_DATE'].unique()
plt.scatter(X2, Y2, color='black', label= 'Canada CPI calculated from average rents by area with a populatio
plt.xlabel('year')
plt.ylabel('Canada CPI')
plt.grid(True)
plt.legend()
plt.show()
```



In [157]:

```
# now lets compare Ottawa-Gatineau area to Canada:
A=CPI_df[(CPI_df['Products and product groups']=='All-items')
       & (CPI_df['city']=='Canada')]

# the base year is 2002
# All-items , Rented accommodation, Shelter
# Canada

X=A['REF_DATE']
Y=A['VALUE']
plt.scatter(X, Y , color='red', label='Canada Consumer Price Index, annual average, not seasonally adjusted'
B=rent_after2002_df[rent_after2002_df['city']=='Ottawa-Gatineau'].groupby(['REF_DATE']).mean()
Y2=B['CPI']
X2=rent_after2002_df['REF_DATE'].unique()
plt.scatter(X2, Y2, color='black', label= 'Ottawa-Gatineau CPI calculated from average rents')
plt.xlabel('year')
plt.ylabel('Canada CPI')
plt.grid(True)
plt.legend()
plt.show()
```