In [1]:

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

In [2]:

```
df=pd.read_csv("C:\\Users\\harsh\\Personal\\A1a,b\\NSSO.csv")
```

In [3]:

```
df.describe()
```

Out[3]:

```
       Unnamed: 0           grp  Round_Centre    FSU_number   Round  Schedule_Number  Sample       Sector   state  State_Region  ...  prepa
count  2041.000000  2.041000e+03        2041.0   2041.000000  2041.0           2041.0  2041.0  2041.000000  2041.0   2041.000000  ...
mean  85732.754532  7.362489e+31           1.0  73626.030867    68.0             10.0     1.0     1.187653     2.0     21.485546  ...
 std  22639.858394  1.162820e+31           0.0  11623.665973     0.0              0.0     0.0     0.390531     0.0      0.499914  ...
 min  38448.000000  4.940000e+31           1.0  49410.000000    68.0             10.0     1.0     1.000000     2.0     21.000000  ...
 25%  95876.000000  7.910000e+31           1.0  79115.000000    68.0             10.0     1.0     1.000000     2.0     21.000000  ...
 50%  96386.000000  7.920000e+31           1.0  79179.000000    68.0             10.0     1.0     1.000000     2.0     21.000000  ...
 75%  96896.000000  7.920000e+31           1.0  79243.000000    68.0             10.0     1.0     1.000000     2.0     22.000000  ...
 max  98126.000000  7.950000e+31           1.0  79488.000000    68.0             10.0     1.0     2.000000     2.0     22.000000  ...

8 rows × 383 columns
```

In [4]:

```
import pandas as pd

column_names = df.columns.tolist()
print(column_names)
```

```
['Unnamed: 0', 'grp', 'Round_Centre', 'FSU_number', 'Round', 'Schedule_Number', 'Sample', 'Sector', 'state', 'State_
Region', 'District', 'Stratum_Number', 'Sub_Stratum', 'Schedule_type', 'Sub_Round', 'Sub_Sample', 'FOD_Sub_Region',
'Hamlet_Group_Sub_Block', 'Second', 'X_Stage_Stratum', 'HHS_No', 'Level', 'Filler', 'hhdsz', 'NIC_2008', 'NCO_2004',
'HH_type', 'Religion', 'Social_Group', 'Whether_owns_any_land', 'Type_of_land_owned', 'Land_Owned', 'Land_Leased_i
n', 'Otherwise_possessed', 'Land_Leased_out', 'Land_Total_possessed', 'During_July_June_Cultivated', 'During_July_Ju
ne_Irrigated', 'NSS', 'NSC', 'MLT', 'land_tt', 'Cooking_code', 'Lighting_code', 'Dwelling_unit_code', 'Regular_salar
y_earner', 'Perform_Ceremony', 'Meals_seved_to_non_hhld_members', 'Possess_ration_card', 'Type_of_ration_card', 'MPC
E_URP', 'MPCE_MRP', 'Person_Srl_No', 'Relation', 'Sex', 'Age', 'Marital_Status', 'Education', 'Days_Stayed_away', 'N
o_of_Meals_per_day', 'Meals_School', 'Meals_Employer', 'Meals_Others', 'Meals_Payment', 'Meals_At_Home', 'Item_Cod
e', 'Source_Code', 'ricepds_q', 'riceos_q', 'ricetotal_q', 'chira_q', 'khoi_q', 'muri_q', 'ricepro_q', 'riceGT_q',
'Wheatpds_q', 'wheatos_q', 'wheattotal_q', 'maida_q', 'suji_q', 'sewai_q', 'bread_q', 'wheatp_q', 'wheatGT_q', 'jowa
rp_q', 'bajrap_q', 'maizep_q', 'barleyp_q', 'milletp_q', 'ragip_q', 'cerealot_q', 'cerealtot_q', 'cerealsub_q', 'cer
ealstt_q', 'arhar_q', 'gramdal_q', 'gramwholep_q', 'gramGT_q', 'moong_q', 'masur_q', 'urd_q', 'peasdal_q', 'khesari_
q', 'otpulse_q', 'gramp_q', 'besan_q', 'pulsep_q', 'pulsestot_q', 'pulsestt_q', 'soyabean_q', 'milk_q', 'babyfood_
q', 'milkcond_q', 'curd_q', 'ghee_q', 'butter_q', 'icecream_q', 'otmilkp_q', 'Milktotal_q', 'milkprott_q', 'vanas_
q', 'musoil_q', 'gnoil_q', 'cocooil_q', 'edioilothr_q', 'edibletotal_q', 'ediblest_q', 'eggsno_q', 'fishprawn_q', 'g
oatmeat_q', 'beef_q', 'pork_q', 'chicken_q', 'othrbirds_q', 'nonvegtotal_q', 'emftt_q', 'potato_q', 'onion_q', 'tama
to_q', 'brinjal_q', 'radish_q', 'carrot_q', 'palak_q', 'chillig_q', 'bhindi_q', 'parwal_q', 'cauli_q', 'cabbage_q',
'pumpkin_q', 'peas_q', 'fbeans_q', 'lemonno_q', 'otveg_q', 'vegtt_q', 'bananano_q', 'jackfruit_q', 'watermel_q', 'pi
neaplno_q', 'cocono_q', 'cocogno_q', 'guava_q', 'sighara_q', 'orangeno_q', 'papayar_q', 'mango_q', 'kharbooz_q', 'pe
ars_q', 'berries_q', 'leechi_q', 'apple_q', 'grapes_q', 'otfruits_q', 'fruitstt_q', 'fruitt_total', 'cocodf_q', 'gnu
tdf_q', 'datesdf_q', 'cashewdf_q', 'walnutdf_q', 'otnutsdf_q', 'kishmish_q', 'otherdf_q', 'dryfruitstotal_q', 'dftt_
q', 'sugarpds_q', 'sugaros_q', 'sugarst_q', 'gur_q', 'misri_q', 'honey_q', 'sugartotal_q', 'sugartt_q', 'salt_q', 'g
inger_q', 'garlic_q', 'jeera_q', 'dhania_q', 'turnmeric_q', 'blackpepper_q', 'drychilly_q', 'tamarind_q', 'currypowd
er_q', 'oilseeds_q', 'spicesothr_q', 'spicetot_q', 'spicestotal_q', 'teacupno_q', 'tealeaf_q', 'teatotal_q', 'cofeen
o_q', 'coffeepwdr_q', 'cofeetotal_q', 'ice_q', 'coldbvrg_q', 'juice_q', 'othrbevrg_q', 'bevergest_q', 'Biscuits_q',
'preparedsweet_q', 'pickle_q', 'sauce_jam_q', 'Othrprocessed_q', 'Beveragestotal_q', 'ricepds_v', 'riceos_v', 'ricet
otal_v', 'chira_v', 'khoi_v', 'muri_v', 'ricepro_v', 'riceGT_v', 'Wheatpds_v', 'wheatos_v', 'wheattotal_v', 'maida_
v', 'suji_v', 'sewai_v', 'bread_v', 'wheatp_v', 'wheatGT_v', 'jowarp_v', 'bajrap_v', 'maizep_v', 'barleyp_v', 'mille
tp_v', 'ragip_v', 'cerealot_v', 'cerealtot_v', 'cerealsub_v', 'cerealstt_v', 'arhar_v', 'gramdal_v', 'gramwholep_v',
'gramGT_v', 'moong_v', 'masur_v', 'urd_v', 'peasdal_v', 'khesari_v', 'otpulse_v', 'gramp_v', 'besan_v', 'pulsep_v',
'pulsestot_v', 'pulsestt_v', 'soyabean_v', 'milk_v', 'babyfood_v', 'milkcond_v', 'curd_v', 'ghee_v', 'butter_v', 'ic
ecream_v', 'otmilkp_v', 'Milktotal_v', 'milkprott_v', 'vanas_v', 'musoil_v', 'gnoil_v', 'cocooil_v', 'edioilothr_v',
'edibletotal_v', 'ediblest_v', 'eggsno_v', 'fishprawn_v', 'goatmeat_v', 'beef_v', 'pork_v', 'chicken_v', 'othrbirds_
v', 'nonvegtotal_v', 'emftt_v', 'potato_v', 'onion_v', 'tamato_v', 'brinjal_v', 'radish_v', 'carrot_v', 'palak_v',
'chillig_v', 'bhindi_v', 'parwal_v', 'cauli_v', 'cabbage_v', 'pumpkin_v', 'peas_v', 'fbeans_v', 'lemonno_v', 'otveg_
v', 'vegtt_v', 'bananano_v', 'jackfruit_v', 'watermel_v', 'pineaplno_v', 'cocono_v', 'cocogno_v', 'guava_v', 'sighar
a_v', 'orangeno_v', 'papayar_v', 'mango_v', 'kharbooz_v', 'pears_v', 'berries_v', 'leechi_v', 'apple_v', 'grapes_v',
'otfruits_v', 'fruitstt_v', 'cocodf_v', 'gnutdf_v', 'datesdf_v', 'cashewdf_v', 'walnutdf_v', 'otnutsdf_v', 'kishmish
_v', 'otherdf_v', 'dryfruitstotal_v', 'dftt_v', 'sugarpds_v', 'sugaros_v', 'sugarst_v', 'gur_v', 'misri_v', 'honey_
v', 'sugartotal_v', 'sugartt_v', 'salt_v', 'ginger_v', 'garlic_v', 'jeera_v', 'dhania_v', 'turnmeric_v', 'blackpeppe
r_v', 'drychilly_v', 'tamarind_v', 'currypowder_v', 'oilseeds_v', 'spicesothr_v', 'spicetot_v', 'spicestotal_v', 'te
acupno_v', 'tealeaf_v', 'teatotal_v', 'cofeeno_v', 'coffeepwdr_v', 'cofeetotal_v', 'ice_v', 'coldbvrg_v', 'juice_v',
'othrbevrg_v', 'bevergest_v', 'Biscuits_v', 'preparedsweet_v', 'pickle_v', 'sauce_jam_v', 'Othrprocessed_v', 'Bevera
gestotal_v', 'foodtotal_v', 'foodtotal_q', 'state_1', 'Region', 'fruits_df_tt_v', 'fv_tot']
```

In [5]:

```python
#Get the number of rows and columns
num_rows, num_cols = df.shape
# Print the results
print("Number of rows:", num_rows)
print("Number of columns:", num_cols)
```

```
Number of rows: 2041
Number of columns: 384
```

In [6]:

```
print('Column Name \t # of Blanks:')
for col in df.columns:
    if df[col].isna().sum()>0:
        print(col,'\t',df[col].isna().sum())
```

```
Column Name      # of Blanks:
NIC_2008         151
NCO_2004         148
HH_type          1
Type_of_land_owned       278
Land_Owned       290
Land_Leased_in   1650
Otherwise_possessed      1987
Land_Leased_out          1950
During_July_June_Cultivated      547
During_July_June_Irrigated       1630
Meals_seved_to_non_hhld_members          28
Type_of_ration_card      208
Days_Stayed_away         1293
Meals_School     2033
Meals_Employer   2012
Meals_Others     1560
Meals_Payment    1903
Meals_At_Home    25
Source_Code      5
soyabean_q       2041
soyabean_v       2041
```

In [7]:

```
for col in df.columns:
    if df[col].isna().sum()>0:
        df[col]=df[col].fillna(df[col].mean())
```

In [8]:

```
df[col].isna().sum()
```

Out[8]:

```
0
```

In [9]:

```
 def find_outliers_IQR(df):

     q1=df.quantile(0.25)
     q3=df.quantile(0.75)
     IQR=q3-q1
#outliers = df[((df<(q1-1.5*IQR)) | (df>(q3+1.5*IQR)))]
     return q1,q3,IQR
```

In [10]:

```
num_cols = []
for col in df.columns:
    if df[col].dtypes=='int64' or df[col].dtypes == 'float64':
        num_cols.append(col)
```

In [11]:

```
 print('Column Name \t # of Outliers')
for col in num_cols:
    q1,q3,IQR=find_outliers_IQR(df[col])
    #print(q1,q3,IQR)
    q1=q1.astype(float)
    q3=q3.astype(float)
    IQR=IQR.astype(float)
    no_of_outliers = df[((df[col]<(q1-1.5*IQR)) | (df[col]>(q3+1.5*IQR)))].shape[0]
    if no_of_outliers>0:
        print(col+':\t'+str(no_of_outliers))
```

```
jeera_v:        80
dhania_v:       118
turnmeric_v:    105
blackpepper_v:  330
drychilly_v:    108
tamarind_v:     38
oilseeds_v:     4
spicesothr_v:   67
spicetot_v:     122
spicestotal_v:  122
Biscuits_v:     115
preparedsweet_v:        129
pickle_v:       480
sauce_jam_v:    43
Othrprocessed_v:        46
Beveragestotal_v:       160
foodtotal_v:    83
foodtotal_q:    115
fruits_df_tt_v: 140
fv_tot: 135
```

In [12]:

```
df['Sector'] = df['Sector'].replace({1: 'urban', 2: 'rural'})
df['Sector'].unique()
```

Out[12]:

```
array(['rural', 'urban'], dtype=object)
```

In [13]:

```
 df['District']=df['District'].replace({ 1: 'Kangra',
2: 'Shimla',
3: 'Mandi',
4: 'Kullu',
5: 'Chamba',
6: 'Hamirpur',
7: 'Solan',
8: 'Lahaul & Spiti',
9: 'Simraur',
10: 'Una',
11: 'Bilaspur',
12: 'Kinnaur'
})
df['District'].unique()
```

Out[13]:

```
array(['Bilaspur', 'Lahaul & Spiti', 'Simraur', 'Solan', 'Una',
       'Hamirpur', 'Kangra', 'Chamba', 'Kullu', 'Shimla', 'Kinnaur',
       'Mandi'], dtype=object)
```

In [14]:

```
region_stats = df.groupby('Region').agg({
'ricetotal_q': 'mean',
'fruitt_total': 'median',
'Beveragestotal_q': 'max',
'sugartotal_q': 'sum',
})
region_stats
```

Out[14]:

	ricetotal_q	fruitt_total	Beveragestotal_q	sugartotal_q
Region				
1	4.718839	8.44	0.0005	1683.239409
2	4.522898	12.00	0.0005	1527.194935

In [15]:

```python
 # Group by district and calculate summary statistics for critical variables
district_stats = df.groupby('District').agg({
'ricetotal_q': 'mean',
'fruitt_total': 'median',
'Beveragestotal_q': 'max',
'sugartotal_q': 'sum',
})
district_stats
```

Out[15]:

	ricetotal_q	fruitt_total	Beveragestotal_q	sugartotal_q
District				
Bilaspur	4.915768	16.250000	0.00050	343.989329
Chamba	5.015437	1.125000	0.00050	344.756919
Hamirpur	4.966589	19.807692	0.00050	327.980411
Kangra	5.109518	8.785714	0.00050	238.547212
Kinnaur	5.638895	18.250000	0.00050	112.777857
Kullu	5.109361	6.166667	0.00025	157.402601
Lahaul & Spiti	4.294856	9.888889	0.00000	229.177291
Mandi	5.924628	8.128571	0.00050	85.729405
Shimla	5.386070	10.240000	0.00050	508.736881
Simraur	3.767878	11.105000	0.00040	283.150552
Solan	2.241228	14.000000	0.00020	344.362597
Una	3.470081	9.775000	0.00025	233.823290

In [16]:

```python
 # Sort districts based on consumption of ricetotal_q variable
sorted_districts = district_stats.sort_values('ricetotal_q', ascending=False)
sorted_districts
```

Out[16]:

	ricetotal_q	fruitt_total	Beveragestotal_q	sugartotal_q
District				
Mandi	5.924628	8.128571	0.00050	85.729405
Kinnaur	5.638895	18.250000	0.00050	112.777857
Shimla	5.386070	10.240000	0.00050	508.736881
Kangra	5.109518	8.785714	0.00050	238.547212
Kullu	5.109361	6.166667	0.00025	157.402601
Chamba	5.015437	1.125000	0.00050	344.756919
Hamirpur	4.966589	19.807692	0.00050	327.980411
Bilaspur	4.915768	16.250000	0.00050	343.989329
Lahaul & Spiti	4.294856	9.888889	0.00000	229.177291
Simraur	3.767878	11.105000	0.00040	283.150552
Una	3.470081	9.775000	0.00025	233.823290
Solan	2.241228	14.000000	0.00020	344.362597

In [17]:

```python
 top_three_districts = sorted_districts.head(3)
top_three_districts
```

Out[17]:

	ricetotal_q	fruitt_total	Beveragestotal_q	sugartotal_q
District				
Mandi	5.924628	8.128571	0.0005	85.729405
Kinnaur	5.638895	18.250000	0.0005	112.777857
Shimla	5.386070	10.240000	0.0005	508.736881

In [18]:

```
 bottom_three_districts = sorted_districts.tail(3)
bottom_three_districts
```

Out[18]:

	ricetotal_q	fruitt_total	Beveragestotal_q	sugartotal_q
District				
Simraur	3.767878	11.105	0.00040	283.150552
Una	3.470081	9.775	0.00025	233.823290
Solan	2.241228	14.000	0.00020	344.362597

In [19]:

```
 # Print the summary statistics and top/bottom districts
print("Region-wise summary statistics for critical variables:")
print(region_stats)
print("\nDistrict-wise summary statistics for critical variables:")
print(district_stats)
print("\nTop three districts with highest ricetotal_q consumption:")
print(top_three_districts)
print("\nBottom three districts with lowest ricetotal_q consumption:")
print(bottom_three_districts)
```

```
Region-wise summary statistics for critical variables:
        ricetotal_q  fruitt_total  Beveragestotal_q  sugartotal_q
Region
1          4.718839          8.44            0.0005   1683.239409
2          4.522898         12.00            0.0005   1527.194935

District-wise summary statistics for critical variables:
                ricetotal_q  fruitt_total  Beveragestotal_q  sugartotal_q
District
Bilaspur           4.915768     16.250000           0.00050    343.989329
Chamba             5.015437      1.125000           0.00050    344.756919
Hamirpur           4.966589     19.807692           0.00050    327.980411
Kangra             5.109518      8.785714           0.00050    238.547212
Kinnaur            5.638895     18.250000           0.00050    112.777857
Kullu              5.109361      6.166667           0.00025    157.402601
Lahaul & Spiti     4.294856      9.888889           0.00000    229.177291
Mandi              5.924628      8.128571           0.00050     85.729405
Shimla             5.386070     10.240000           0.00050    508.736881
Simraur            3.767878     11.105000           0.00040    283.150552
Solan              2.241228     14.000000           0.00020    344.362597
Una                3.470081      9.775000           0.00025    233.823290

Top three districts with highest ricetotal_q consumption:
          ricetotal_q  fruitt_total  Beveragestotal_q  sugartotal_q
District
Mandi        5.924628      8.128571            0.0005     85.729405
Kinnaur      5.638895     18.250000            0.0005    112.777857
Shimla       5.386070     10.240000            0.0005    508.736881

Bottom three districts with lowest ricetotal_q consumption:
          ricetotal_q  fruitt_total  Beveragestotal_q  sugartotal_q
District
Simraur      3.767878        11.105           0.00040    283.150552
Una          3.470081         9.775           0.00025    233.823290
Solan        2.241228        14.000           0.00020    344.362597
```

In [20]:

```
 import scipy.stats as stats
# Group data by the variable of interest
group1 = df[df['Beveragestotal_q'] == 'Group 1']['Beveragestotal_v']
group2 = df[df['foodtotal_q'] == 'Group 2']['foodtotal_v']
group3 = df[df['fv_tot'] == 'Group 3']['fruits_df_tt_v']
# Perform one-way ANOVA
f_statistic, p_value = stats.f_oneway(group1, group2, group3)
# Print the test result
if p_value < 0.05:
    print("The differences in means are significant.")
else:
    print("The differences in means are not significant.")
```

```
The differences in means are not significant.

C:\Users\harsh\OneDrive\Desktop\python\lib\site-packages\scipy\stats\_stats_py.py:3861: DegenerateDataWarning: at le
ast one input has length 0
  warnings.warn(stats.DegenerateDataWarning('at least one input '
```

In [21]:

```
from sklearn.linear_model import LinearRegression
from sklearn import datasets, linear_model, metrics
```

In [22]:

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

In [23]:

```
!pip install wget
import wget
```

```
Requirement already satisfied: wget in c:\users\harsh\onedrive\desktop\python\lib\site-packages (3.2)
```

In [24]:

```
url = "<https://github.dev/datta07/INDIAN-SHAPEFILES/blob/master/STATES/HIMACHAL%20PRADESH/HIMACHAL%20PRADESH%20D
```

In [25]:

```
rt requests

= "https://github.dev/datta07/INDIAN-SHAPEFILES/blob/master/STATES/HIMACHAL%20PRADESH/HIMACHAL%20PRADESH%20District%20
_path = "C:/Users/harsh/OneDrive/Desktop/python/HIMACHAL PRADESH District Hq.geojson"  # Path to save the downloaded f

onse = requests.get(url)
 open(file_path, "wb") as file:
file.write(response.content)

t("File downloaded successfully.")
```

```
File downloaded successfully.
```

In [26]:

```
file_path = r"C:\Users\harsh\Deskptop\HIMACHAL PRADESH Hq.geojson"
```

In [27]:

```
!pip install folium
```

```
Requirement already satisfied: folium in c:\users\harsh\onedrive\desktop\python\lib\site-packages (0.14.0)
Requirement already satisfied: jinja2>=2.9 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from folium)
(3.1.2)
Requirement already satisfied: numpy in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from folium) (1.2
3.5)
Requirement already satisfied: branca>=0.6.0 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from foliu
m) (0.6.0)
Requirement already satisfied: requests in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from folium)
(2.28.1)
Requirement already satisfied: MarkupSafe>=2.0 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from jin
ja2>=2.9->folium) (2.1.1)
Requirement already satisfied: charset-normalizer<3,>=2 in c:\users\harsh\onedrive\desktop\python\lib\site-packages
(from requests->folium) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from reques
ts->folium) (3.4)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (from
requests->folium) (2023.5.7)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in c:\users\harsh\onedrive\desktop\python\lib\site-packages (fr
om requests->folium) (1.26.14)
```

In [28]:

```
import geopandas as gpd
import folium
import warnings

warnings.filterwarnings('ignore')
```

In [29]:

```
dist_map_df = gpd.read_file('C:\\Users\\harsh\\OneDrive\\Desktop\\SCMA\\indian_districts\\indian_districts.shp')
```

In [30]:

```
dist_map_df.head()
```

Out[30]:

	latitude	total popu	state name	district n	state nam0	marginal w	main worke	country	iso	district 0	total work	longitude	district c	non-worker	geometry
0	33.184377	0	NaN	NaN	NaN	0	0	India	IND	Mirpur	0	74.320913	NaN	0	POLYGON ((74.34567 33.38107, 74.35369 33.37884...
1	16.720088	31394	Pondicherry	District Yanam	Pondicherry	611	9298	India	IND	Yanam	9909	82.237839	34_01	21485	MULTIPOLYGON (((82.28556 16.69756, 82.25880 16...
2	32.503986	33224	Himachal Pradesh	District Lahul & Spiti	Himachal Pradesh	1879	19209	India	IND	Lahul and Spiti	21088	77.504765	02_03	12136	POLYGON ((76.80274 33.23656, 76.80854 33.24236...
3	28.739873	33363	Arunachal Pradesh	District Upper Siang*	Arunachal Pradesh	1710	15395	India	IND	Upper Siang	17105	94.807556	12_09	16258	POLYGON ((95.26806 28.94682, 95.28233 28.94861...
4	27.662086	38924	Arunachal Pradesh	Distruct Tawang	Arunachal Pradesh	3593	18134	India	IND	Tawang	21727	91.929890	12_01	17197	POLYGON ((92.31591 27.77827, 92.27934 27.67703...

In [31]:

```
HP_map_df = dist_map_df[dist_map_df['state name']=='Himachal Pradesh']
```

In [32]:

```
HP_map_df.head()
```

Out[32]:

	latitude	total popu	state name	district n	state nam0	marginal w	main worke	country	iso	district 0	total work	longitude	district c	non-worker	geometry
2	32.503986	33224	Himachal Pradesh	District Lahul & Spiti	Himachal Pradesh	1879	19209	India	IND	Lahul and Spiti	21088	77.504765	02_03	12136	POLYGON ((76.80274 33.23656, 76.80854 33.24236...
18	31.599252	78334	Himachal Pradesh	District Kinnaur	Himachal Pradesh	7498	40313	India	IND	Kinnaur	47811	78.366013	02_12	30523	POLYGON ((78.90703 31.25939, 78.91550 31.22505...
65	31.321164	340885	Himachal Pradesh	District Bilaspur	Himachal Pradesh	56056	110652	India	IND	Bilaspur H	166708	76.646416	02_08	174177	POLYGON ((76.45575 31.42709, 76.48474 31.39542...
71	31.882023	381571	Himachal Pradesh	District Kullu	Himachal Pradesh	49798	166715	India	IND	Kullu	216513	77.387910	02_04	165058	POLYGON ((77.85087 31.78747, 77.83972 31.78524...
73	31.660133	412700	Himachal Pradesh	District Hamirpur	Himachal Pradesh	85535	119870	India	IND	Hamirpur	205405	76.500571	02_06	207295	POLYGON ((76.71488 31.58810, 76.69079 31.60193...

In [33]:

```python
HP_map_df = HP_map_df[['district n','geometry']]
HP_map_df.head(2)
```

Out[33]:

	district n	geometry
2	District Lahul & Spiti	POLYGON ((76.80274 33.23656, 76.80854 33.24236...
18	District Kinnaur	POLYGON ((78.90703 31.25939, 78.91550 31.22505...

In [34]:

```python
HP_map_df.set_index('district n', inplace=True)
```

In [35]:

```python
df.rename(columns={'District name':'district n'}, inplace=True)
```

In [36]:

```python
HP_map_df.reset_index('district n', inplace=True)
```

In [37]:

```python
print(HP_map_df.columns)

# Correct the column name for capitalizing
HP_map_df = HP_map_df.rename(columns={'district n': 'districtn'})

# Capitalize the values in the 'districtn' column
HP_map_df['districtn'] = HP_map_df['districtn'].str.capitalize()

# Print the first 2 rows of the updated GeoDataFrame
print(HP_map_df.head(2))
```

```
Index(['district n', 'geometry'], dtype='object')
                districtn                                           geometry
0  District lahul & spiti  POLYGON ((76.80274 33.23656, 76.80854 33.24236...
1        District kinnaur  POLYGON ((78.90703 31.25939, 78.91550 31.22505...
```

In [38]:

```
HP_map_df.districtn.unique()
```

Out[38]:

```
array(['District lahul & spiti', 'District kinnaur', 'District bilaspur',
       'District kullu', 'District hamirpur', 'District una',
       'District sirmaur', 'District chamba', 'District solan',
       'District shimla', 'District mandi', 'District kangra'],
      dtype=object)
```

In [42]:

```
# Check the column names of the DataFrame
print(df.columns)

# Correct the column name for accessing unique values
column_name = 'District'  # Replace with the actual column name

# Access the unique values in the column
unique_values = df[column_name].unique()

# Print the unique values
print(unique_values)

```

```
Index(['Unnamed: 0', 'grp', 'Round_Centre', 'FSU_number', 'Round',
       'Schedule_Number', 'Sample', 'Sector', 'state', 'State_Region',
       ...
       'pickle_v', 'sauce_jam_v', 'Othrprocessed_v', 'Beveragestotal_v',
       'foodtotal_v', 'foodtotal_q', 'state_1', 'Region', 'fruits_df_tt_v',
       'fv_tot'],
      dtype='object', length=384)
['Bilaspur' 'Lahaul & Spiti' 'Simraur' 'Solan' 'Una' 'Hamirpur' 'Kangra'
 'Chamba' 'Kullu' 'Shimla' 'Kinnaur' 'Mandi']
```

In [45]:

```python
# Check the column names of the DataFrame
print(df.columns)

# Correct the column name for grouping
column_name = 'District'  # Replace with the actual column name

# Group the DataFrame by the column
grouped_df = df.groupby(column_name)[['fv_tot']].sum()

# Print the grouped DataFrame
print(grouped_df)

```

```
Index(['Unnamed: 0', 'grp', 'Round_Centre', 'FSU_number', 'Round',
       'Schedule_Number', 'Sample', 'Sector', 'state', 'State_Region',
       ...
       'pickle_v', 'sauce_jam_v', 'Othrprocessed_v', 'Beveragestotal_v',
       'foodtotal_v', 'foodtotal_q', 'state_1', 'Region', 'fruits_df_tt_v',
       'fv_tot'],
      dtype='object', length=384)
                      fv_tot
District
Bilaspur        29985.918343
Chamba          18344.879244
Hamirpur        19785.838526
Kangra          14811.199316
Kinnaur         10815.541905
Kullu            8402.695222
Lahaul & Spiti  16267.991970
Mandi            5180.360714
Shimla          34089.206861
Simraur         27305.972605
Solan           17605.419896
Una             16486.194965
```

In [51]:

```
1 fig, ax = plt.subplots(figsize=(10, 6))
2 ax.axis('off')
3 ax.set_title('Himachal Consumption district', fontdict={'fontsize': '25', 'fontweight': '3'})
4
5 merged.plot(column='Total_c', cmap='Wistia', linewidth=0.8, ax=ax, edgecolor='0.8', legend=True)
6
7 plt.show()
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
Cell In[51], line 5
      2 ax.axis('off')
      3 ax.set_title('Himachal Consumption district', fontdict={'fontsize': '25', 'fontweight': '3'})
----> 5 merged.plot(column='Total_c', cmap='Wistia', linewidth=0.8, ax=ax, edgecolor='0.8', legend=True)
      7 plt.show()

AttributeError: 'ellipsis' object has no attribute 'plot'
```

Himachal Consumption district

In [ ]:

```
```