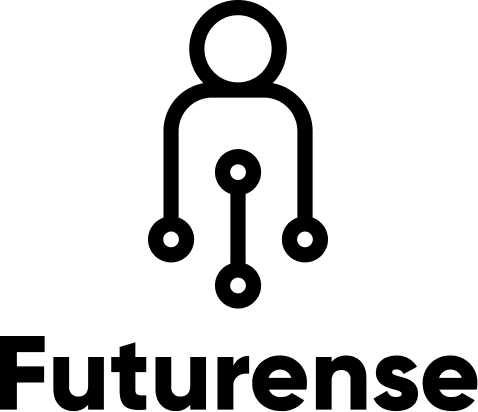
**Futurense Technologies**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

****

**Project Report**

**Group 5**

**Names:**

**1. Akshay R**

**2. Bhuvana C**

**3. Neell Ravindra Ambere**

**4. Pratham Chopra**

**ABSTRACT** C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

This project, titled "Clean the Census Data and Extract Insights from Housing Data," is dedicated to refining the 2011 census data to enhance its applicability in healthcare. Leveraging Python, specifically utilizing Pandas, NumPy, and Matplotlib, our approach involved a systematic curation of the dataset. This encompassed the judicious selection of pivotal columns, standardization of terminology, and meticulous handling of missing data.

The resulting cleaned census data, neatly stored as "census.csv," establishes a robust foundation for upcoming healthcare analyses. Beyond data refinement, our project extends its impact by integrating insights derived from housing data. This augmentation not only enriches the dataset but also provides a nuanced and comprehensive perspective for future healthcare research endeavors. By adopting an iterative and thorough process, our project ensures the accuracy and depth required for meaningful explorations within the realm of healthcare analytics.

**Contents**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

1. ABSTRACT
2. LIST OF FIGURES/CODES/GRAPHS
3. PROBLEM STATEMENT
4. OBJECTIVE
5. DESCRIPTION
6. CODE
7. SCREENSHOT
8. RESULTS AND CONCLUSION
9. FUTURE SCOPE
10. REFERENCES

**List of Figures/Codes/Graphs**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

Please Refer The Screenshot Section.Image 1 : Problem Statement - Keeping The Relevant Data

Image 2 : Problem Statement 1 - Contd.

Image 3 : Problem Statement 2 - Rename The Column Names

Image 4 : Problem Statement - Rename State / UT Names

Image 5 : Problem Statement - New State / UT Formation

Image 6 : Problem Statement 5 - Find & Process Missing Data

Image 7 : Graph - Comparison Of Missing Data Before & After Filling

Image 8 : Problem Statement 6 - Save Data

Image 9 : Problem Statement 7 - Process The Relevant Data

Image 10 : Problem Statement 7 - Contd.

**Problem Statements**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

1. Clean the Census Data:

* Select and retain relevant columns for healthcare analysis.

1. Rename the Column Names:

* Standardize column names for uniformity, considering the census year.

1. Rename State/UT Names:

* Achieve uniformity in State/UT names for cohesive dataset representation.

1. New State/UT Formation:

* Update state names post Telangana and Ladakh formations for historical accuracy.

1. Find and Process Missing Data:

* Identify and fill in missing data, and visualize improvements before and after data filling.

1. Save Data:

* Save processed data in "census.csv" for easy access in the "Clean\_Data" folder.

1. Process Relevant Data from Housing Data:

* Extract and calculate absolute values from housing data, and save processed data in "housing.csv." Check for data consistency across datasets.

**Objective**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

In our project, we improved healthcare-focused census data by refining columns, standardizing names, and addressing missing data. We updated state and union territory names, reflecting real-world changes. The processed census data was, saved as "census.csv,”. Insights from housing data were integrated, forming a comprehensive foundation for healthcare analyses.

**Description**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

* 1. Data Selection:

Identified essential columns, including population, literacy rates, and household details.

Pandas function is utilized for efficient extraction and organization of relevant data.

* 2. Column Standardization:

Employed Pandas functions to standardize column names, ensuring clarity and uniformity.

Key libraries used: Pandas, and NumPy for numerical operations.

* 3. State and Union Territory Uniformization:

Custom Python functions applied for consistent formatting of state and union territory names.

Considered exceptions like "and" for proper capitalization.

* 4. Geopolitical Changes Integration:

Custom Python functions updated state names post-Telangana (2014) and Ladakh (2019) formations.

* 5. Data Cleaning and Visualization:

Pandas functions are used for filtering and organizing columns, retaining only essential information.

Matplotlib is employed to create visualizations, comparing missing data percentages before and after the data-filling process.

* 6. Data Export:

Pandas functions are utilized for exporting the refined census data, saved as "census.csv" for future healthcare analyses.

* 7. Housing Data Integration:

Functions employed for reading, processing, and seamlessly integrating housing data into the census dataset.

**Code**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

**CLEAN THE CENSUS DATA**

**Problem Statement 1: (Keep the relevant data)**  
import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

# Load the census data

census\_data\_relevant = pd.read\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/census\_2011.csv")

census\_data\_relevant

# Select the relevant columns

census\_data\_relevant = census\_data\_relevant[["State name", "District name", "Population", "Male", "Female", "Literate", "Male\_Literate", "Female\_Literate", "Rural\_Households", "Urban\_Households", "Households","Age\_Group\_0\_29", "Age\_Group\_30\_49", "Age\_Group\_50", "Age not stated"]]

census\_data\_relevant

**Problem Statement 2: (Rename the Column names)**

# Rename the column names

census\_data\_relevant = census\_data\_relevant.rename(columns={"State name": "State/UT", "District name": "District", "Male\_Literate": "Literate\_Male", "Female\_Literate": "Literate\_Female","Rural\_Households": "Households\_Rural", "Urban\_Households": "Households\_Urban", "Age\_Group\_0\_29": "Young\_and\_Adult", "Age\_Group\_30\_49": "Middle\_Aged", "Age\_Group\_50": "Senior\_Citizen", "Age not stated": "Age\_Not\_Stated"})

census\_data\_relevant

**Problem Statement 3: (Rename State/UT Names)**

# Function to format state names

def format\_state\_name(state\_name):

return " ".join(word.title() for word in state\_name.split()).replace(" And ", " and ")

# Apply the function to the State/UT column

census\_data\_relevant["State/UT"] = census\_data\_relevant["State/UT"].apply(format\_state\_name)

census\_data\_relevant.to\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/clean data/census.csv", index=False)

census\_data\_relevant

**Problem Statement 4: (New State/UT formation)**

telangana\_districts\_file = "/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/Telangana.txt"

with open(telangana\_districts\_file, 'r') as file:

telangana\_districts = [line.strip() for line in file]

# Update State/UT for Telangana

census\_data\_relevant.loc[census\_data\_relevant['District'].isin(telangana\_districts), 'State/UT'] = 'Telangana'

# Read Ladakh districts

ladakh\_districts = ["Leh","Kargil","Leh(Ladakh)"]

# Update State/UT for Ladakh

census\_data\_relevant.loc[census\_data\_relevant['District'].isin(ladakh\_districts), 'State/UT'] = 'Ladakh'

# Save the updated DataFrame

census\_data\_relevant.to\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/clean data/cleandata.csv", index=False)

census\_data\_relevant

**Problem Statement 5: (Find and process Missing Data)**

# Calculate the percentage of missing data before filling

missing\_percentage\_before = pd.read\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/census\_2011.csv").isnull().mean() \* 100

# Fill missing values based on hints

census\_data\_relevant["Population"].fillna(census\_data\_relevant["Male"] + census\_data\_relevant["Female"], inplace=True)

census\_data\_relevant["Literate"].fillna(census\_data\_relevant["Literate\_Male"] + census\_data\_relevant["Literate\_Female"], inplace=True)

census\_data\_relevant["Population"].fillna(census\_data\_relevant["Young\_and\_Adult"] + census\_data\_relevant["Middle\_Aged"] + census\_data\_relevant["Senior\_Citizen"] + census\_data\_relevant["Age\_Not\_Stated"], inplace=True)

census\_data\_relevant["Households"].fillna(census\_data\_relevant["Households\_Rural"] + census\_data\_relevant["Households\_Urban"], inplace=True)

# Calculate percentage of missing data after filling

missing\_percentage\_after = census\_data\_relevant.isnull().mean() \* 100

# Visualization

fig, ax = plt.subplots(figsize=(10, 6))

sns.barplot(x=missing\_percentage\_before.index, y=missing\_percentage\_before, color='red', label='Before Filling')

sns.barplot(x=missing\_percentage\_after.index, y=missing\_percentage\_after, color='green', label='After Filling')

ax.set\_ylabel('Percentage of Missing Data')

ax.set\_title('Comparison of Missing Data Before and After Filling')

ax.legend()

plt.xticks(rotation=45, ha='right')

plt.tight\_layout()

plt.show()

**Problem Statement 6: (Save Data)**

census\_data\_relevant.to\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/clean data/census.csv", index=False)

census\_data\_relevant

**Extract Information from the housing data along with the census data**

**Problem Statement 7: (Process the relevant data)**

# Read housing data

housing\_data = pd.read\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/housing.csv")

# Read census data

census\_data = pd.read\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/clean data/census.csv")

housing\_data

# Function to format state names

def format\_state\_name(state\_name):

return " ".join(word.title() for word in state\_name.split()).replace(" And ", " and ")

# Apply the function to the State/UT column

housing\_data["State Name"] = housing\_data["State Name"].apply(format\_state\_name)

housing\_data.to\_csv("/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/housing.csv", index=False)

housing\_data

# Merge housing and census data on District Name

merged\_data = pd.merge(housing\_data, census\_data, left\_on='District Name', right\_on='District', how='outer')

# Calculate absolute values for relevant columns

merged\_data['Households\_Rural'] = merged\_data['Total Number of households'] \* merged\_data['Households\_Rural'] / 100

merged\_data['Households\_Urban'] = merged\_data['Total Number of households'] \* merged\_data['Households\_Urban'] / 100

merged\_data['Households\_Rural\_Livable'] = merged\_data['Households\_Rural'] \* merged\_data['Total Number of Livable'] / 100

merged\_data['Households\_Urban\_Livable'] = merged\_data['Households\_Urban'] \* merged\_data['Total Number of Livable'] / 100

merged\_data['Households\_Rural\_Dilapidated'] = merged\_data['Households\_Rural'] \* merged\_data['Total Number of Dilapidated'] / 100

merged\_data['Households\_Urban\_Dilapidated'] = merged\_data['Households\_Urban'] \* merged\_data['Total Number of Dilapidated'] / 100

merged\_data['Households\_Rural\_Toilet\_Premise'] = merged\_data['Households\_Rural'] \* merged\_data['Latrine\_premise'] / 100

merged\_data['Households\_Urban\_Toilet\_Premise'] = merged\_data['Households\_Urban'] \* merged\_data['Latrine\_premise'] / 100

# Save the processed data

housing\_data\_merged="/Users/bcr/Downloads/DS - Python + SQL + ETL - Healthcare Census (4)/DS - Python + SQL + ETL - Healthcare Census/Data/clean data/housing.csv"

merged\_data.to\_csv(housing\_data\_merged, index=False)

# Report if any district data is in one file but not the other

missing\_districts\_census = census\_data[~census\_data['District'].isin(housing\_data['District Name'])]['District']

missing\_districts\_housing = housing\_data[~housing\_data['District Name'].isin(census\_data['District'])]['District Name']

print("Districts in census but not in housing data: \n ", missing\_districts\_census.tolist())

print("\n Districts in housing but not in census data: \n ", missing\_districts\_housing.tolist())

**Screenshot**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

Image 1 : Problem Statement - Keeping The Relevant Data

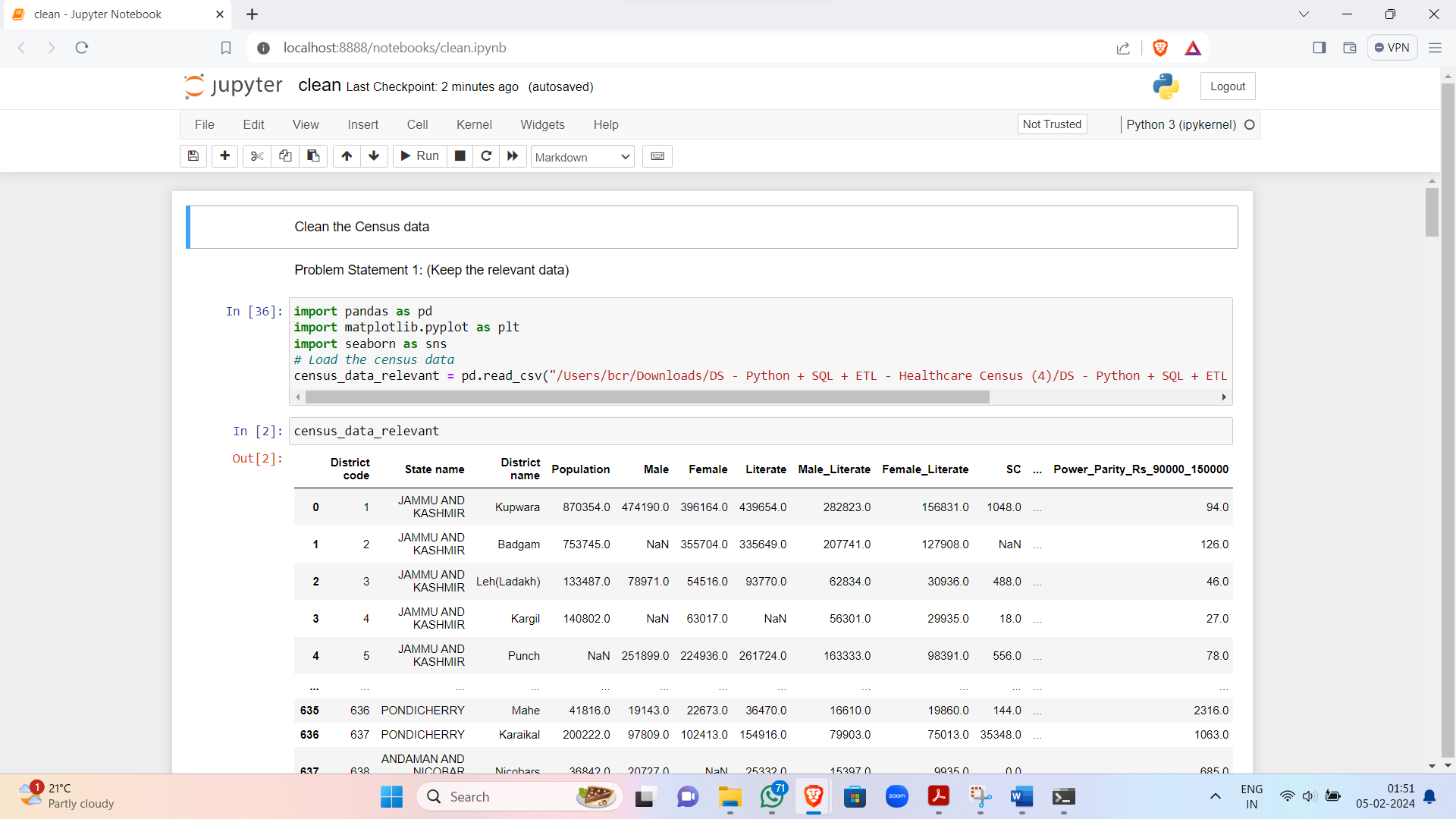
****

Image 2 : Problem Statement 1 - Contd.

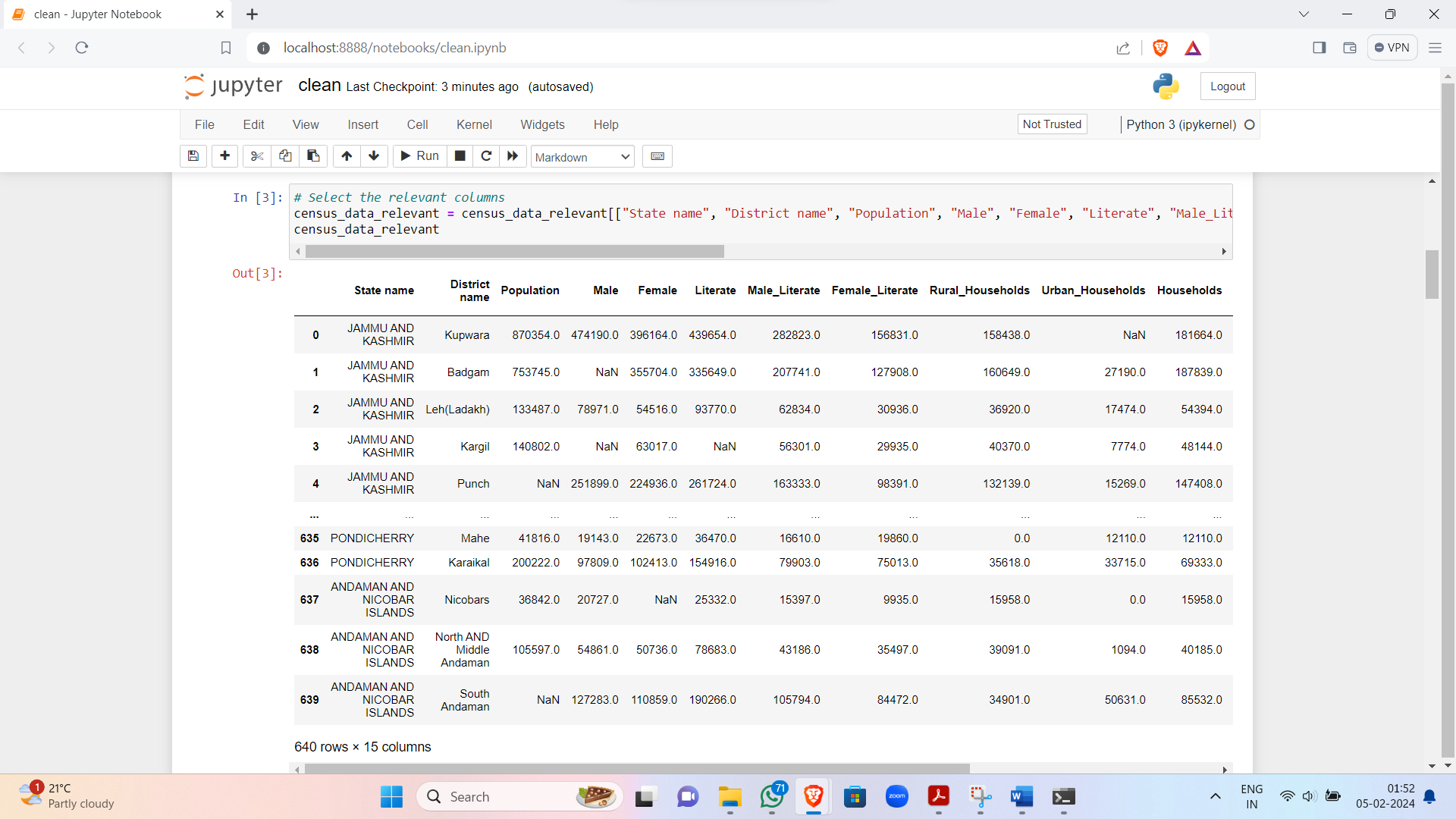
****

Image 3 : Problem Statement 2 - Rename The Column Names

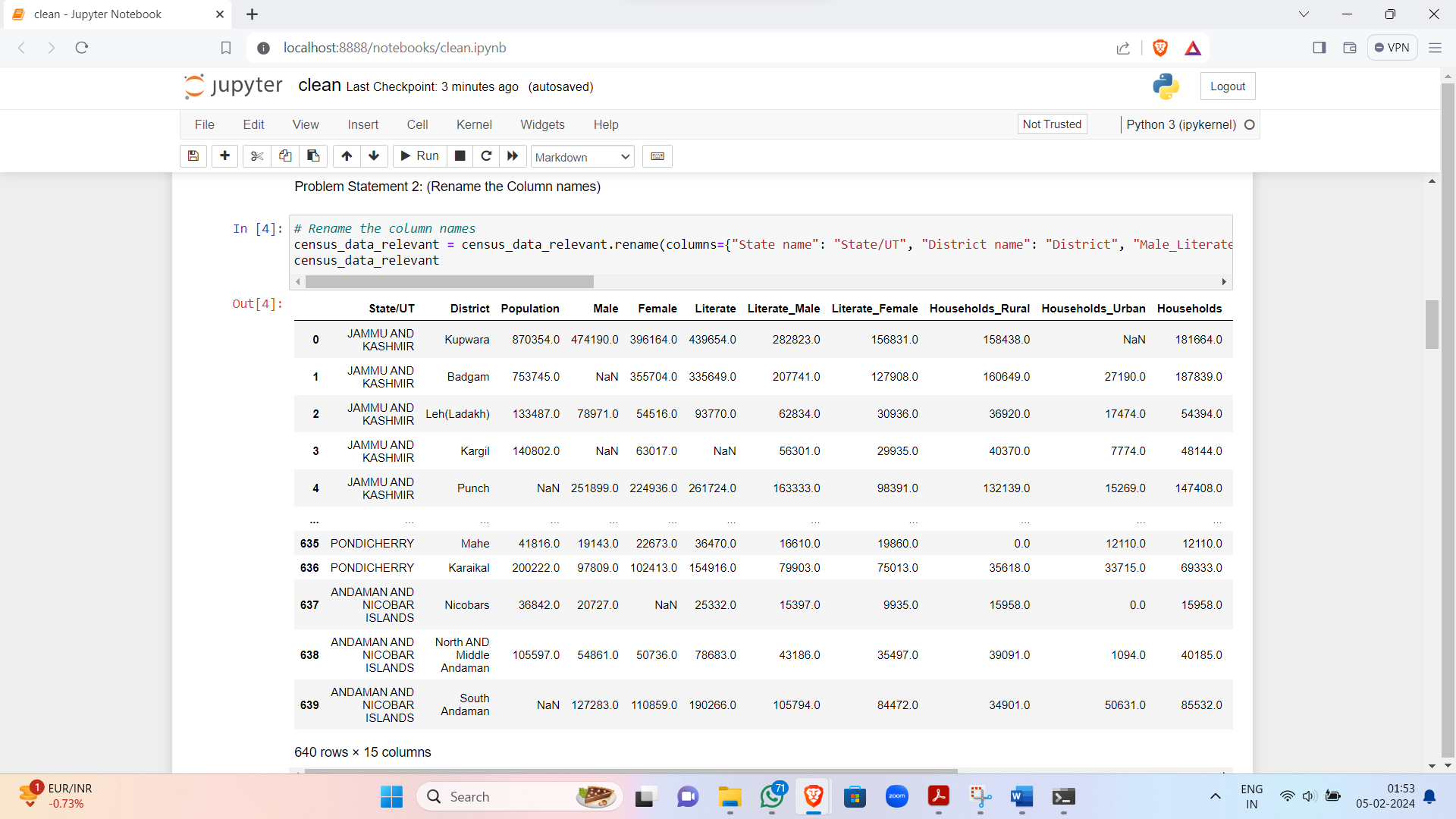


Image 4 : Problem Statement - Rename State / UT Names

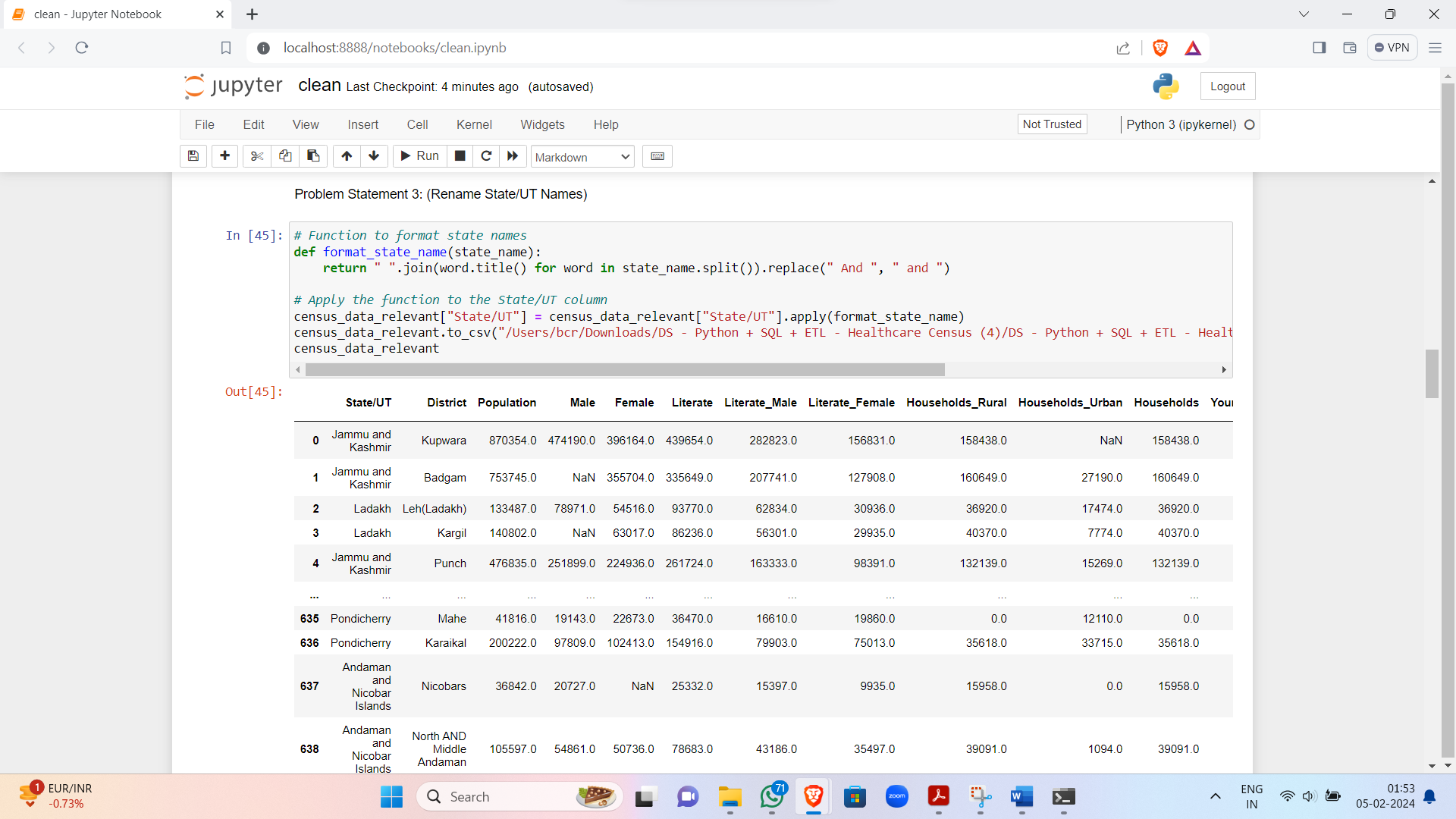


Image 5 : Problem Statement - New State / UT Formation

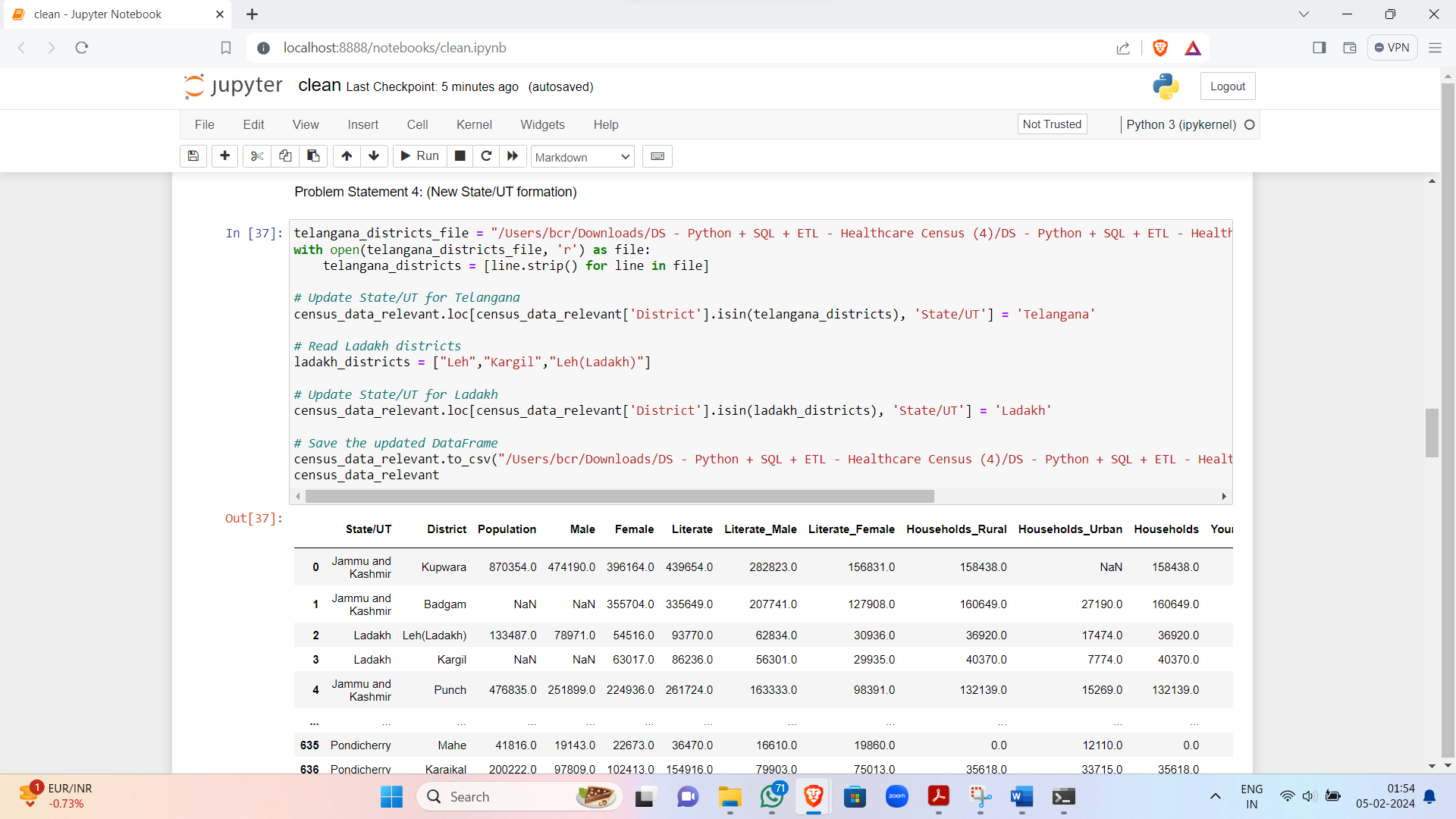


Image 6 : Problem Statement 5 - Find & Process Missing Data

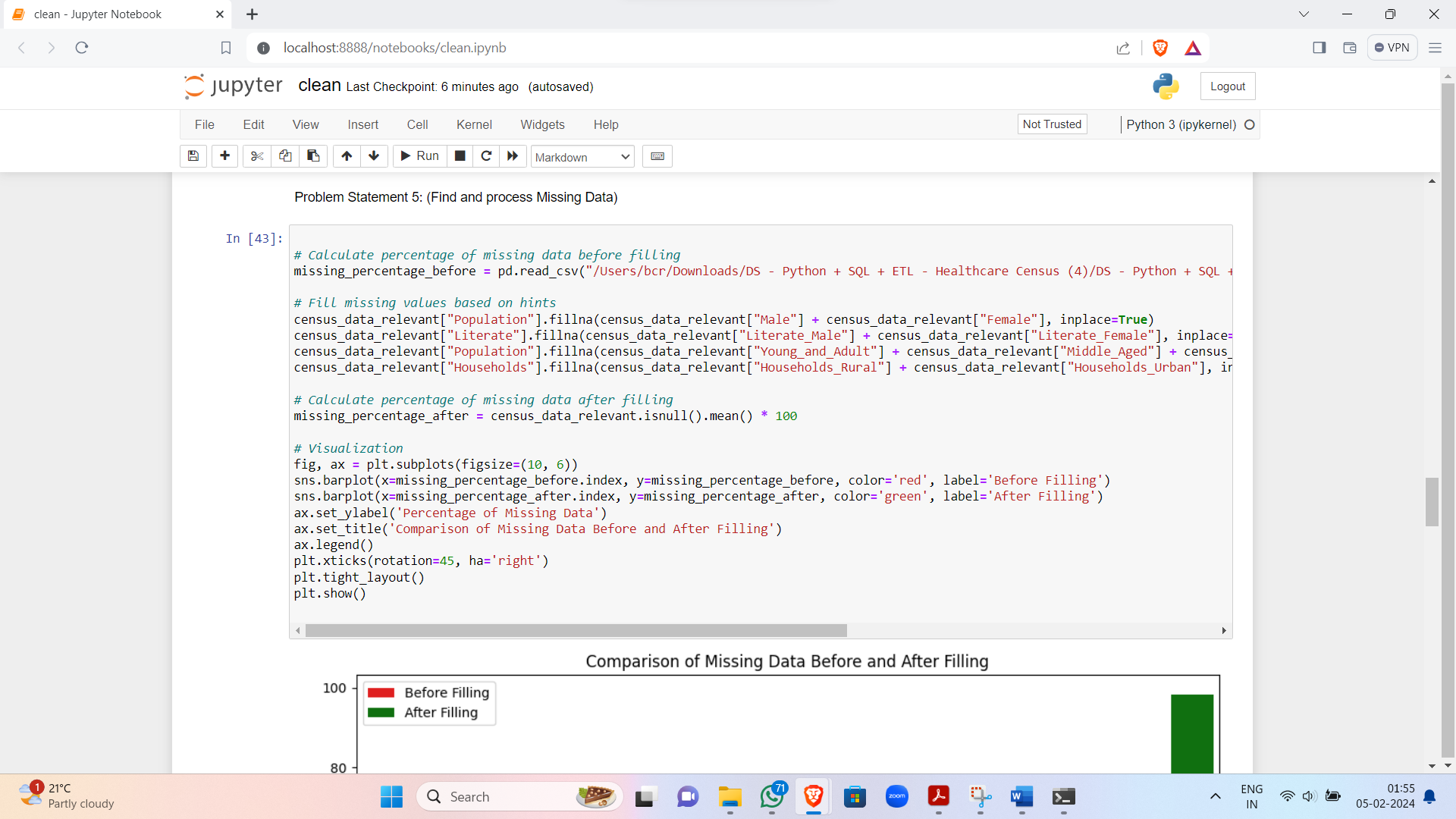


Image 7 : Graph - Comparison Of Missing Data Before & After Filling

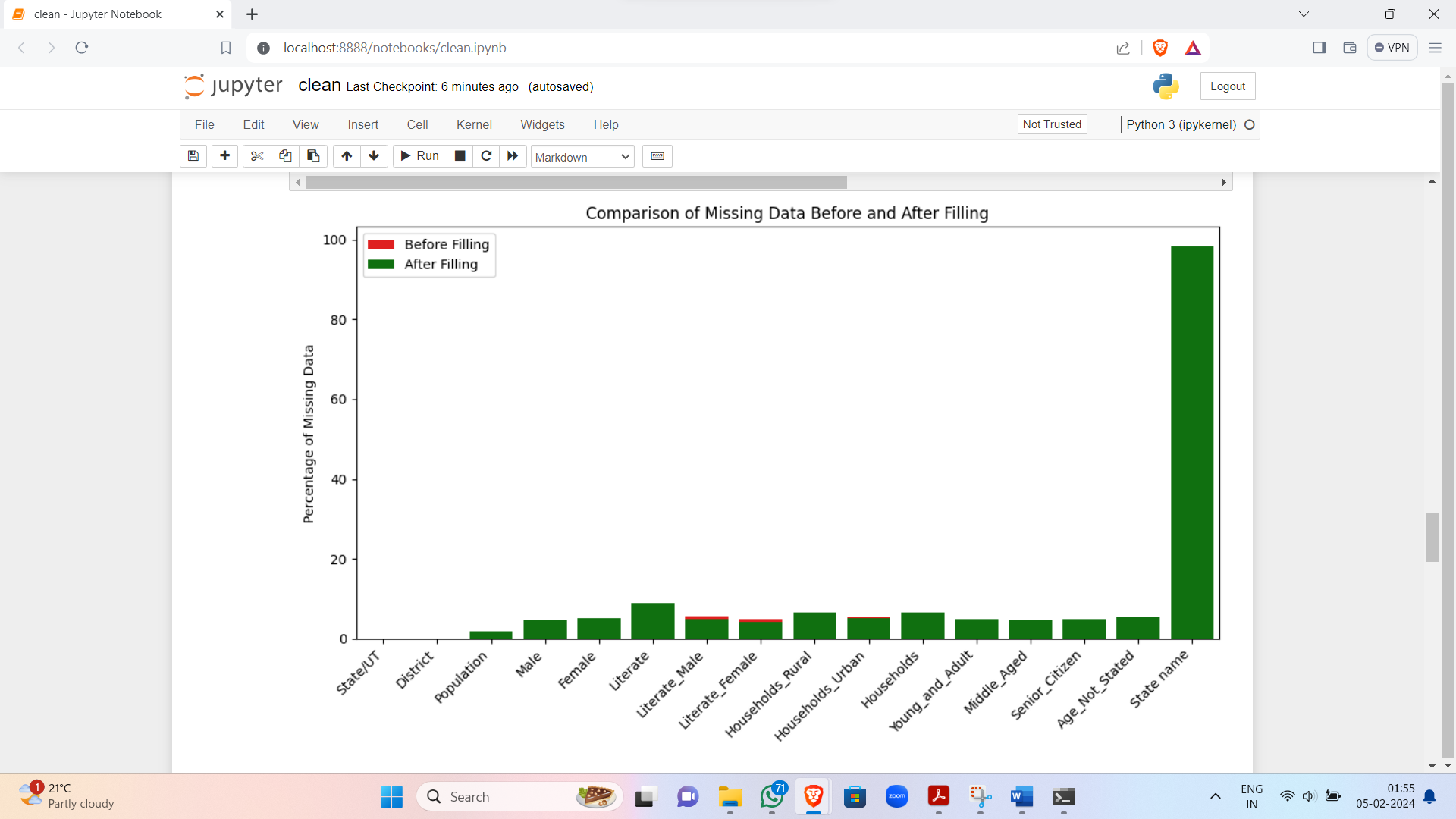


Image 8 : Problem Statement 6 - Save Data

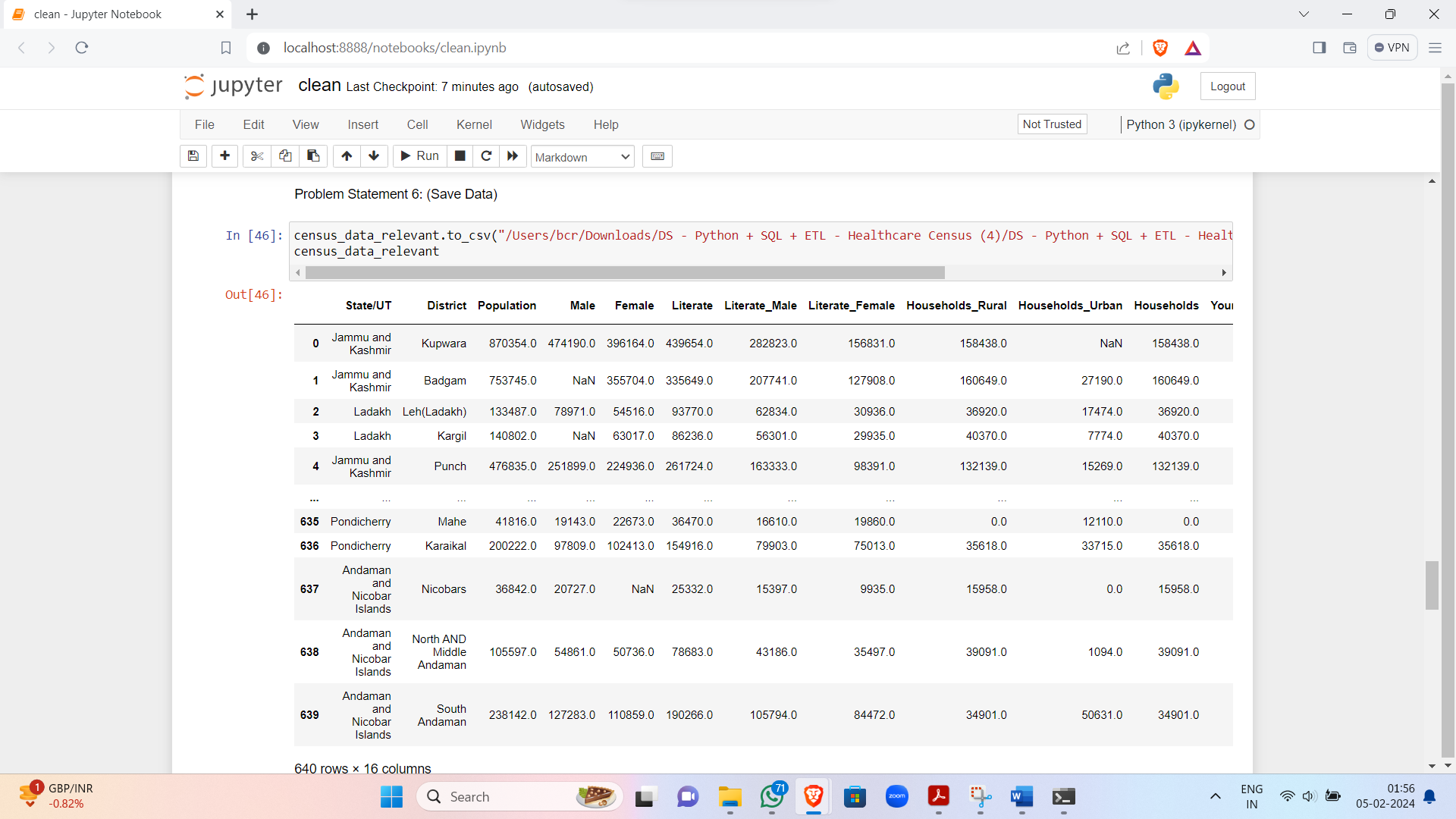


Image 9 : Problem Statement 7 - Process The Relevant Data

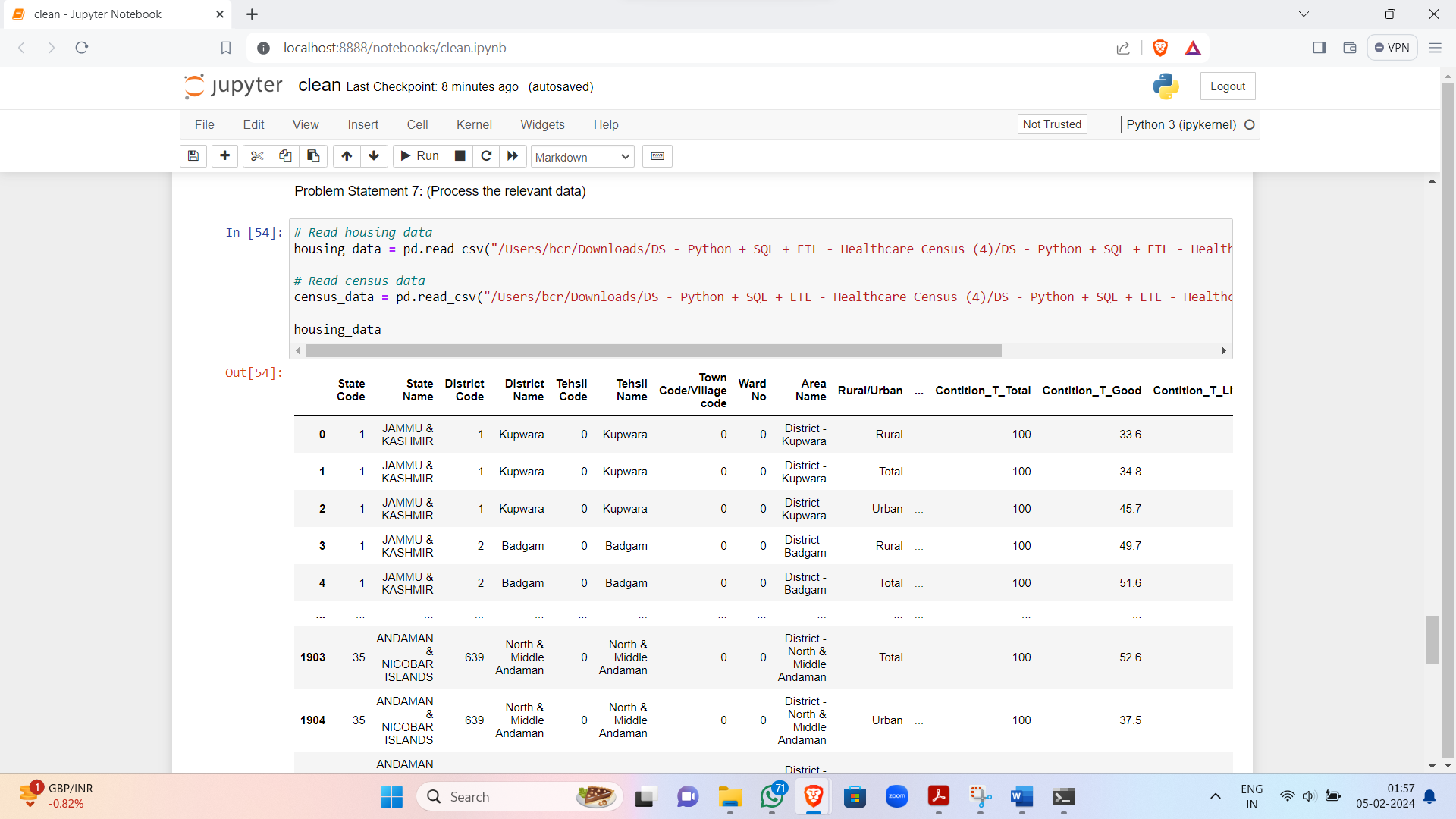
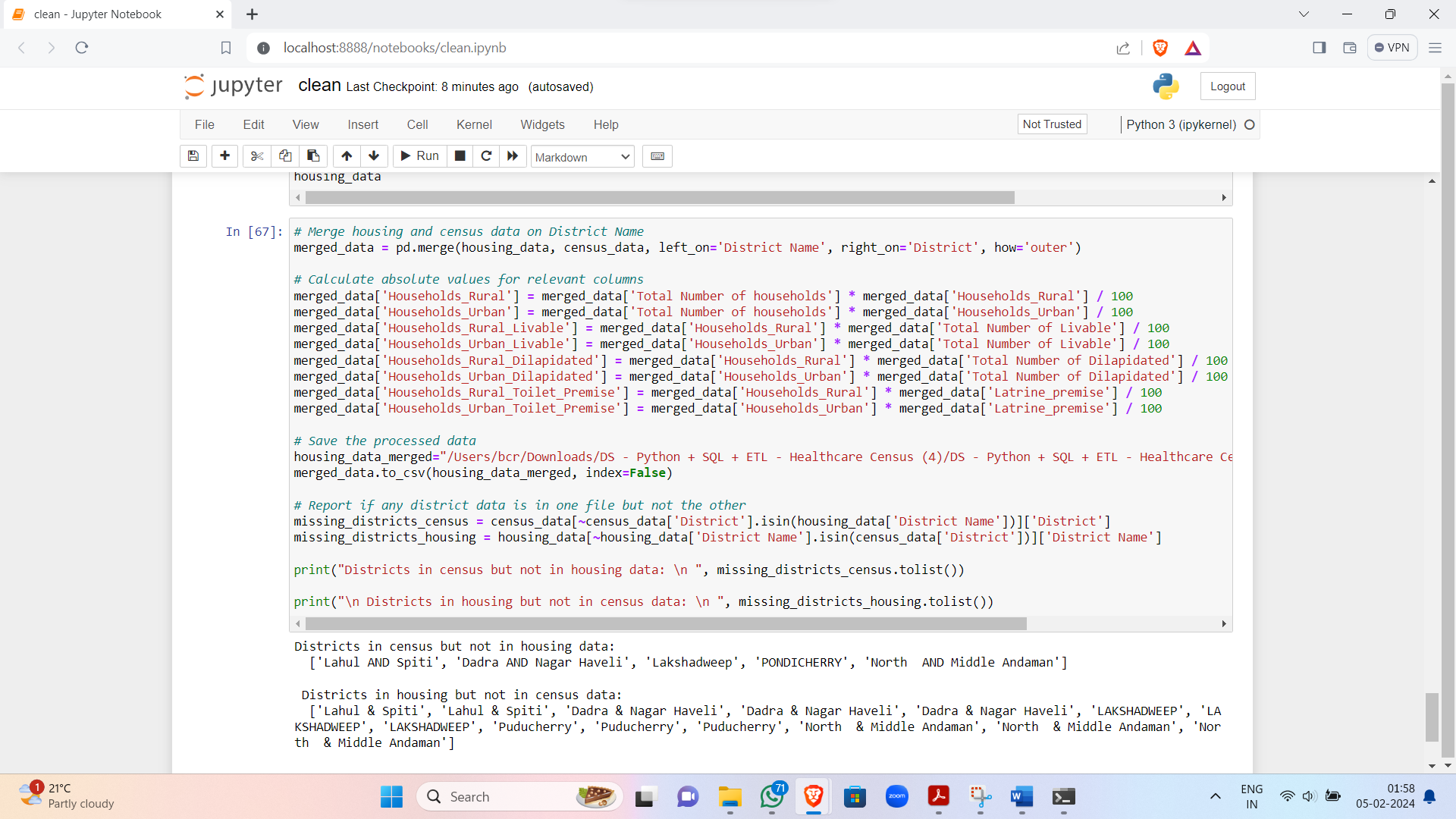


Image 10 : Problem Statement 7 - Contd.



**Results and Conclusion**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

* 1. Refining the Data:

We worked on refining the dataset, focusing on essential aspects like population, literacy rates, and household details.

* 2. Consistent Naming:

Successfully made the dataset more organized by standardizing column and state/union territory names.

* 3. Handling Changes:

Integrated changes from Telangana (2014) and Ladakh (2019) formations seamlessly.

* 4. Complete Data Set:

Addressed missing values, resulting in a much more complete dataset for thorough healthcare analysis.

* 5. Visual Insight:

Created visuals comparing missing data percentages, providing a clear before-and-after view.

* 6. Easy Accessibility:

Saved the refined census data as "census.csv," making it easily accessible for future healthcare analyses.

* 7. Holistic Foundation:

Extended our analysis to housing data, enriching the dataset for comprehensive healthcare research.

* 8. Project Conclusion:

Successfully achieved our goals, creating a clean, standardised dataset ready for valuable insights in healthcare research.

**Future Scope**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

* 1. Implementation of Remaining Problem Statements:

Execute the pending 20 problem statements, enhancing the dataset comprehensively and addressing diverse healthcare aspects.

* 2. Advanced Data Visualization for Housing Insights:

Develop intuitive visualizations for housing data, illustrating metrics like households per 100 people, toilet facilities, and urban-to-rural population ratios.

* 3. Extended Comparative Analyses - Hospitals:

Expand analyses by integrating insights from hospital data, aiding Aliah in identifying states requiring urgent additional hospital beds.

* 4. Efficient Header Management for Hospital Data:

Streamline hospital data headers using automated functions, ensuring uniformity and easy integration with census and housing datasets.

* 5. Comprehensive Healthcare Disparity Analysis:

Investigate healthcare facility disparities, recommending new government hospitals based on the least bed-to-population ratios and governmental resources.

**References**C:\Users\hp\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\B69B6BB2.tmp

* Pandas Documentation. (n.d.). Retrieved from <https://pandas.pydata.org/pandas-docs/stable/index.html>
* Matplotlib Documentation. (n.d.). Retrieved from <https://matplotlib.org/stable/contents.html>
* NumPy Documentation. (n.d.). Retrieved from <https://numpy.org/doc/stable/>
* OpenAI Generative Pre-Trained Model - 3.5. (n.d.). Retrieved from<https://chat.openai.com/>
* Google Bard AI Experiment. (n.d.). Retrieved from <https://bard.google.com/chat>
* Jupyter Notebook Documentation. (n.d.). Retrieved from <https://jupyter.org/documentation>
* Stack Overflow Community. (n.d.). Retrieved from<https://stackoverflow.com/>
* GeeksforGeeks. (n.d.). Retrieved from<https://www.geeksforgeeks.org/>
* Data Cleaning and Preprocessing with Python. (n.d.). Retrieved from <https://towardsdatascience.com/data-cleaning-with-python-and-pandas-detecting-missing-values-3e9c6ebcf78b>