```python
import pandas as pd
import numpy as np
import scipy.stats as stats

def load_data(file_path):
    try:
        data = pd.read_csv(file_path, header=None)
        data = data.dropna()
        return data[0].values
    except Exception as e:
        print(f"Error loading data: {e}")
        return []

def estimate_population_mean(data, sample_size, confidence_level):
    if sample_size > len(data):
        raise ValueError("Sample size is larger than available data.")

    sample = np.random.choice(data, size=sample_size, replace=False)
    sample_mean = np.mean(sample)
    sample_std = np.std(sample, ddof=1)  # Sample standard deviation

    # Get z or t critical value
    t_crit = stats.t.ppf((1 + confidence_level) / 2, df=sample_size - 1)

    margin_of_error = t_crit * (sample_std / np.sqrt(sample_size))
    confidence_interval = (sample_mean - margin_of_error, sample_mean + margin_of_error)

    return sample_mean, margin_of_error, confidence_interval
if __name__ == "__main__":
    file_path = (r'C:\Users\91637\OneDrive\Desktop\sev\rare_elements.csv')
    data = load_data(file_path)

    if len(data) == 0:
        print("No data loaded. Please check the file.")
    else:
        try:
            sample_size = int(input("Enter sample size: "))
            confidence_level = float(input("Enter confidence level (e.g., 0.95): "))
            precision = float(input("Enter desired level of precision (margin of error): "))

            sample_mean, margin_of_error, confidence_interval = estimate_population_mean(
                data, sample_size, confidence_level
            )

            print(f"\nSample Mean: {sample_mean:.4f}")
            print(f"Margin of Error: ±{margin_of_error:.4f}")
            print(f"{int(confidence_level*100)}% Confidence Interval: {confidence_interval}")

            if margin_of_error <= precision:
                print("✅ Desired level of precision achieved.")
            else:
                print("⚠️ Desired level of precision not achieved. Consider increasing the sample size.")

        except ValueError as e:
            print(f"Invalid input: {e}")
```

OUTPUT

Enter sample size: 10
Enter confidence level (e.g., 0.95): 0.94
Enter desired level of precision (margin of error): 1
Sample Mean: 2.5114
Margin of Error: ±0.8571
94% Confidence Interval: (np.float64(1.6543328761780742), np.float64(3.368563685330212))
✅ Desired level of precision achieved.