Exploratory tasks to be performed on the X dataset of your choice from column C:

In [165]:

```
# First we will import the required libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import math
import scipy.stats as st
from scipy.stats import norm,poisson,binom,geom,gamma
from sklearn.preprocessing import StandardScaler
from scipy import stats
import warnings
import seaborn as sns
import plotly.express as px
import plotly.graph_objects as go
import plotly.io as pio
pio.templates.default = "plotly_dark"
warnings.filterwarnings('ignore')
%matplotlib inline
```

Our X dataset is a dataset which contains data about staff shortage during covid and number of beds occupied by covid patients

In [70]:

```
df = pd.read_csv('COVID-19_Reported_Patient_Impact_and_Hospital_Capacity_by_State_Times

```

In [71]:

```
df_x = df.loc[df['state'].isin(['KS', 'IL'])]
df_x['date']= pd.to_datetime(df_x['date'])
df_x_final=df_x.sort_values(['state', 'date'])
df_x_final.reset_index(drop=True, inplace=True)
```

In [73]:

```
covid_data = df_x_final[['state','date','critical_staffing_shortage_today_yes','inpatie
covid_data.head()
```

Out[73]:

	state	date	critical_staffing_shortage_today_yes	inpatient_beds	inpatient_beds_used	inpatie
0	IL	2020-02-18	0	36.0	20.0	
1	IL	2020-02-19	0	86.0	17.0	
2	IL	2020-02-20	0	86.0	13.0	
3	IL	2020-02-21	0	86.0	14.0	
4	IL	2020-02-22	0	86.0	14.0	

In [76]:

```
df_x_IL = covid_data[covid_data['state'] == 'IL']
```

In [141]:

```
df_x_KS = covid_data[covid_data['state'] == 'KS']
```

In [95]:

```
dff = (df_x_IL[(df_x_IL['date']>pd.to_datetime('2020-10-31')) & (df_x_IL['date']<pd.to_
```

In [98]:

```
dfc = (df_case_IL[(df_case_IL['submission_date']>pd.to_datetime('2020-10-31')) & (df_ca
```

In [167]:

```
# We are using correlation to compare the data between per day cases and Critical staff
```

In [168]:

```python
# helper function to compute correlation
def computeCorrelation(x,y):
    x_mean = np.mean(x)
    y_mean = np.mean(y)
    print(x_mean)
    print(y_mean)
    xy = 0
    xi_x = 0
    yi_y = 0

    for i in range(len(x)):
        xy += ((x[i]-x_mean) * (y[i] - y_mean))
        xi_x += np.square(x[i] - x_mean)
        yi_y += np.square(y[i] - y_mean)

    return xy/((np.sqrt(xi_x * yi_y)))
```

In [121]:

```python
x = dff['critical_staffing_shortage_today_yes']
y = dfc['per_day_cases']
correlation = computeCorrelation(x,y)
#validate it with corr

print("Our Function Correlation: %1.3f " % (correlation))

```

```
37.49122807017544
9146.491228070176
Our Function Correlation: 0.843
```

```
# From the above data we can see a very high correlation between staff shortage and per day cases
```

In [134]:

```python
x = np.array(dff['inpatient_beds_used_covid'])
y = np.array(dfc['per_day_cases'])
correlation = computeCorrelation(x,y)
#validate it with corr

print("Our Function Correlation: %1.3f " % (correlation))

```

```
5234.666666666667
9146.491228070176
Our Function Correlation: 0.529
```

From the above data we can see a very high correlation between inpatient beds occupied by covid patients and per day cases

Part2: Now we try to check the occuring of second wave in August 2022 by taking prewave and postwave data and applying permutation test

In [135]:

```
df_prewave= (df_case_IL[(df_case_KS['submission_date']>pd.to_datetime('2021-05-24')) &
```

In [136]:

```
(df_case_KS['submission_date']>pd.to_datetime('2021-07-30')) & (df_case_KS['submission_date
```

In [137]:

```
def permutation_test(X, Y, n, threshold):
    T_obs = abs(np.mean(X) - np.mean(Y))
    print(T_obs, np.mean(X), np.mean(Y))
    xy = np.append(X,Y)
#     xy.info()
    p_value = 0.0
    for i in range(n):
        permutation = np.random.permutation(xy)
        X1 = permutation[:len(X)]
        Y1 = permutation[len(X):]
        Ti = abs(np.mean(X1) - np.mean(Y1))
        if(Ti > T_obs):
            p_value += 1.0
#         print(p_value, T_obs, Ti)
    p_value = p_value/n
    print("The p-value is: ", p_value)
    if(p_value <= threshold):
        print("==> Reject the Null Hypothesis")
    else:
        print("==> Accept the Null Hypothesis")
    return
```

In [169]:

```
# PERMUTATION TEST: Hypotheses and Results
print("--------------------------------------------------------------------------------

permutation_test(np.array(df_postwave['per_day_deaths']),np.array(df_prewave['per_day_d
print(np.mean(np.array(df_postwave['per_day_deaths'])))
print(np.mean(np.array(df_prewave['per_day_deaths'])))
print("--------------------------------------------------------------------------------

permutation_test(np.array(df_postwave['per_day_cases']),np.array(df_prewave['per_day_ca
print("--------------------------------------------------------------------------------

```

```
----------------------------------------------------------------------------
-------------------------
2.0333333333333314 17.866666666666667 19.9
The p-value is:  0.494
==> Accept the Null Hypothesis
17.866666666666667
19.9
----------------------------------------------------------------------------
-------------------------
2721.7666666666664 3170.133333333333 448.3666666666667
The p-value is:  0.0
==> Reject the Null Hypothesis
----------------------------------------------------------------------------
-------------------------
```

From the above we see that - the number of cases in increased after wave so permutation test rejects null hypothesis

But number of deaths remained steady. So permutations test accepts null hypotheseis

In [ ]:

```

```