Health Care Data Analysis

By Preston Givens

Introduction

I worked on a Health Care Data set that was compromised of Health care data from India. I will be evaluating the Healthcare, illnesses, and Under Age Marriages for the states in the country.

Annual Health Survey Dataset: https://www.kaggle.com/datasets/rajanand/key-indicators-of-annual-health-survey?resource=download (https://www.kaggle.com/datasets/rajanand/key-indicators-of-annual-health-survey?resource=download)

Questions asked when looking through the Dataset:

1. What State in India had the highest total mean marriages for females?

2. Which State in India had the most people suffering from fever per 100,000 people?

3. Which State in India had the most females suffering from Tuberculosis per 100,000 people?

4. Which State in India had the highest total mean age of Males and females drop out of school between ages 6-17?

5. Which State in India had the most males sick from chronic illnesses per 100,000 people?

Imported Libraries

In [1]:
```
#imported Libraries used in Notebook
import pandas as pd
import numpy as np

#Visualization libraries
import seaborn as sns
import matplotlib.pyplot as plt

from scipy.stats import normaltest
import scipy.stats as stats
#set default seaborn theme
sns.set()

#display graphs and save them in Notebook
%matplotlib inline

#code to hide warnings
import warnings
warnings.filterwarnings("ignore")
```

Importing Data

Load the CSV data into dataframe

In [2]:
```
#read Health_data into dataframe
import pandas as pd
Health1=pd.read_csv('Key_indicator_districtwise.csv')
#Display Dataframe
Health1
```

Out[2]:

	State_Name	EE_Mean_Age_At_Marriage_Female_Total	FF_Children_Currently_Attending_School_
0	Assam	21.70	
1	Assam	21.70	
2	Assam	22.60	
3	Assam	22.10	
4	Assam	21.50	
...	...	...	
279	Uttarakhand	20.90	
280	Uttarakhand	21.86	
281	Uttarakhand	21.85	
282	Uttarakhand	21.63	
283	Uttarakhand	22.00	

284 rows × 8 columns

In [3]:
```
#display the first 5 rows of dataframe
Health1.head()
```

Out[3]:

	State_Name	EE_Mean_Age_At_Marriage_Female_Total	FF_Children_Currently_Attending_School_Ag
0	Assam	21.7	
1	Assam	21.7	
2	Assam	22.6	
3	Assam	22.1	
4	Assam	21.5	

First 5 rows of Dataframe

In [4]:
```
#inspect dataframe using tail method
Health1.tail()
```

Out[4]:

	State_Name	EE_Mean_Age_At_Marriage_Female_Total	FF_Children_Currently_Attending_School_
279	Uttarakhand	20.90	
280	Uttarakhand	21.86	
281	Uttarakhand	21.85	
282	Uttarakhand	21.63	
283	Uttarakhand	22.00	

Inspect under Dataframe

In [5]:
```
#we can use the info method to get additional info about dataframe
#including datatype of different columns
Health1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284 entries, 0 to 283
Data columns (total 8 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   State_Name
284 non-null    object
 1   EE_Mean_Age_At_Marriage_Female_Total
284 non-null    float64
 2   FF_Children_Currently_Attending_School_Age_6_17_Years_Person_Total
284 non-null    float64
 3   FF_Children_Attended_Before_Drop_Out_Age_6_17_Years_Person_Total
284 non-null    float64
 4   JJ_Persons_Suffering_From_Acute_Illness_Per_100000_Population_Fever_All_Ty
pes_Person_Total  284 non-null    float64
 5   KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Pe
rson_Total        284 non-null    float64
 6   KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Ma
le_Total          284 non-null    float64
 7   KK_Having_Diagnosed_For_Chronic_Illness_Per_100000_Population_Tuberculosis
_Tb_Female_Total  284 non-null    float64
dtypes: float64(7), object(1)
memory usage: 17.9+ KB
```

The info method is used to get additional info about dataframe

In [6]:
```
#Display basic statistical details of Dataframe
Health1.describe()
```

Out[6]:

	EE_Mean_Age_At_Marriage_Female_Total	FF_Children_Currently_Attending_School_Age_6_17_Y
count	284.00000	
mean	21.06912	
std	1.03728	
min	17.40000	
25%	20.30000	
50%	21.10000	
75%	21.80000	
max	24.20000	

Conclusion: Displaying basic statistical details such as count, mean, standard deviation, etc, of the Health DataFrame.

Data Cleaning And Organizing

- Inconsistent column names (Have inconsistent capitalization and /or bad characters.
- Missing data
- Outliers
- Duplicate rows
- Untidy can prevent us from transforming our datasets to one suitable for reporting, to a dataset that is suitable for analysis.
- Need to process columns (processing columns before they can be used for data analysis
- Column types can signal unexpected data values

In [7]:
```
#call the dtypes attribute on the dataframe to get the data types of each colums
print(Health1.dtypes)
```

```
State_Name
object
EE_Mean_Age_At_Marriage_Female_Total
float64
FF_Children_Currently_Attending_School_Age_6_17_Years_Person_Total
float64
FF_Children_Attended_Before_Drop_Out_Age_6_17_Years_Person_Total
float64
JJ_Persons_Suffering_From_Acute_Illness_Per_100000_Population_Fever_All_Types_P
erson_Total    float64
KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Person_
Total          float64
KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Male_To
tal            float64
KK_Having_Diagnosed_For_Chronic_Illness_Per_100000_Population_Tuberculosis_Tb_F
emale_Total    float64
dtype: object
```

Data types of each column

In [8]:
```
#IF you have many columns call the column attribute of dataframe
#might find an extra space at the end or beginning of the column name
#or bad characters
#iterating the columns
for col in Health1.columns:
    print(col)
```

```
State_Name
EE_Mean_Age_At_Marriage_Female_Total
FF_Children_Currently_Attending_School_Age_6_17_Years_Person_Total
FF_Children_Attended_Before_Drop_Out_Age_6_17_Years_Person_Total
JJ_Persons_Suffering_From_Acute_Illness_Per_100000_Population_Fever_All_Types_P
erson_Total
KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Person_
Total
KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Population_Male_To
tal
KK_Having_Diagnosed_For_Chronic_Illness_Per_100000_Population_Tuberculosis_Tb_F
emale_Total
```

List of columns that I can call

In [9]:
```
#call the shape attribute to look at the number of rows and columns
Health1.shape
```

Out[9]:
```
(284, 8)
```

Number of rows and columns

In [10]:
```
#Use RanamedColumns to rename columns and change the name
#Change columns names for a better appearance
renamedColumns={'EE_Mean_Age_At_Marriage_Female_Total':'MeanFemales',
                'FF_Children_Attended_Before_Drop_Out_Age_6_17_Years_Person_Total'
                'JJ_Persons_Suffering_From_Acute_Illness_Per_100000_Population_Fev
                'KK_Having_Diagnosed_For_Chronic_Illness_Per_100000_Population_Tub
                'State_Name':'State Name',
                'FF_Children_Currently_Attending_School_Age_6_17_Years_Person_Tota
                'KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Popu
                'KK_Having_Any_Kind_Of_Symptoms_Of_Chronic_Illness_Per_100000_Popu
                }
Health1.rename(columns=renamedColumns, inplace=True)
Health1.head()
```

Out[10]:

	State Name	MeanFemales	Total_Attend_6-17	dropout6-17	Fever100,000	TotalIll100,000	TotalMaleIll100,00
0	Assam	21.7	89.6	10.0	2721.0	11421.0	10800.
1	Assam	21.7	90.7	9.2	2906.0	7497.0	7196.
2	Assam	22.6	85.5	13.2	9987.0	23339.0	21443.
3	Assam	22.1	88.2	10.9	2054.0	10345.0	9843.
4	Assam	21.5	93.8	5.9	2704.0	7585.0	7307.

In [11]:
```
Health1.dtypes
```

Out[11]:
```
State Name              object
MeanFemales            float64
Total_Attend_6-17      float64
dropout6-17            float64
Fever100,000           float64
TotalIll100,000        float64
TotalMaleIll100,000    float64
FemaleTuber100,000     float64
dtype: object
```

Columns renamed

In [12]:
```
#We can convert convert the column to a integer and object type by passing in
#'integer' and object parameter to astype method of the column
Health1['State Name']=Health1['State Name'].astype('object')
Health1['TotalMaleIll100,000']=Health1['TotalMaleIll100,000'].astype('int64')
print (Health1.dtypes)
```

```
State Name              object
MeanFemales            float64
Total_Attend_6-17      float64
dropout6-17            float64
Fever100,000           float64
TotalIll100,000        float64
TotalMaleIll100,000      int64
FemaleTuber100,000     float64
dtype: object
```

In [13]:
```
# Convert 'Total Attending Ages 6-17' to a numeric dtype
Health1['Total_Attend_6-17'] = pd.to_numeric(Health1['Total_Attend_6-17'], errors

# Print the info of tips
print(Health1.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 284 entries, 0 to 283
Data columns (total 8 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   State Name           284 non-null    object
 1   MeanFemales          284 non-null    float64
 2   Total_Attend_6-17    284 non-null    float64
 3   dropout6-17          284 non-null    float64
 4   Fever100,000         284 non-null    float64
 5   TotalIll100,000      284 non-null    float64
 6   TotalMaleIll100,000  284 non-null    int64
 7   FemaleTuber100,000   284 non-null    float64
dtypes: float64(6), int64(1), object(1)
memory usage: 17.9+ KB
None
```

In [14]:
```
#IF you have many columns call the column attribute of dataframe
#might find an extra space at the end or beginning of the column name
#or bad characters
#iterating the columns
for col in Health1.columns:
    print(col)
```

```
State Name
MeanFemales
Total_Attend_6-17
dropout6-17
Fever100,000
TotalIll100,000
TotalMaleIll100,000
FemaleTuber100,000
```

In [15]:
```
Health1.tail()
#Renamed columns makes it easier to locate and find data within a set
```

Out[15]:

	State Name	MeanFemales	Total_Attend_6-17	dropout6-17	Fever100,000	TotalIII100,000	TotalMaleII
279	Uttarakhand	20.90	98.28	1.62	2037.96	14213.40	
280	Uttarakhand	21.86	98.87	1.01	3268.18	9575.36	
281	Uttarakhand	21.85	97.99	1.84	2418.48	12540.22	
282	Uttarakhand	21.63	88.78	10.09	1667.95	6061.21	
283	Uttarakhand	22.00	97.98	1.85	5052.82	11465.69	

Renamed Columns

In [16]:
```
Health1.describe()
#Renamed columns make it easier to interpret columns
```

Out[16]:

	MeanFemales	Total_Attend_6-17	dropout6-17	Fever100,000	TotalIII100,000	TotalMaleIII100,000
count	284.00000	284.000000	284.000000	284.00000	284.000000	284.000000
mean	21.06912	90.327711	7.833275	4699.38412	10181.680634	9126.985915
std	1.03728	5.753271	4.727091	3398.68664	5690.858927	5115.122066
min	17.40000	71.280000	0.600000	587.00000	1940.000000	1897.000000
25%	20.30000	86.690000	3.900000	2371.06750	5764.940000	5151.000000
50%	21.10000	91.425000	6.835000	3579.46500	8901.670000	8116.500000
75%	21.80000	94.607500	10.925000	6068.30500	12946.472500	11715.000000
max	24.20000	99.000000	23.440000	22034.12000	36037.000000	33411.000000

Data Visualization

Data Visualization is used to display statistics and analytics than can project a potential outlook for a specific calculation. Data Visualization projects a potential outlook and helps analysts make data driven decisions.

Question 1: 1. What State in India had the highest total mean marriages for females?

In [17]:
```
#I used pivot table to display the Index for State Name and the Total Mean Marric
#Using the Pivot Table see that Assam is the top State with the total age for mar
Health12=pd.pivot_table(Health1,index='State Name',values='MeanFemales')
Health12
```

Out[17]:

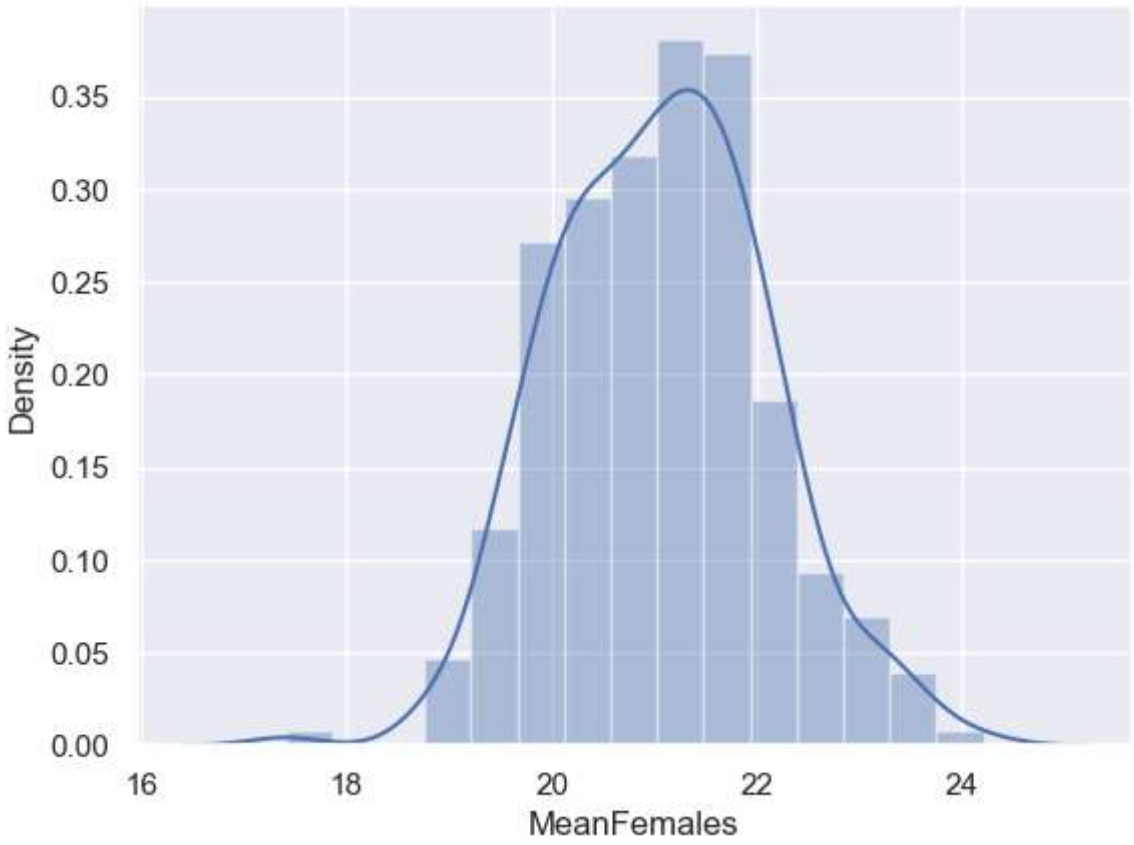
State Name	MeanFemales
Assam	22.047826
Bihar	20.162162
Chhattisgarh	21.183750
Jharkhand	20.600000
Madhya Pradesh	20.704444
Odisha	21.939333
Rajasthan	20.125000
Uttar Pradesh	21.442429
Uttarakhand	21.995385

Conclusion for Pivot Table: Assam was the top State in India with the total age for marriage for women at 22 years old.

Plot of Pivot Table Results

In [18]:
```
# let us see how the Total Mean Marriage Age for Females is distributed
#Using displot to display the results
sns.distplot(Health1['MeanFemales'])
```

Out[18]: <AxesSubplot:xlabel='MeanFemales', ylabel='Density'>



Conclusion: The displot shows how the mean is around the 21-22 age range which confirms my

pivot table analysis.

Groupby() and Describe() for analysis

In [19]:
```
#Display State Name and Fever100,000 for descriptive results on columns
Health21=Health1[['State Name','MeanFemales']]
Health21.groupby('State Name').describe()
```

Out[19]:

	MeanFemales							
	count	mean	std	min	25%	50%	75%	max
State Name								
Assam	23.0	22.047826	0.773345	20.20	21.6500	22.100	22.6000	23.60
Bihar	37.0	20.162162	0.602011	18.80	19.8000	20.200	20.4000	21.60
Chhattisgarh	16.0	21.183750	0.376738	20.37	21.0325	21.270	21.3675	21.90
Jharkhand	18.0	20.600000	0.793355	19.50	20.0250	20.250	21.3000	21.90
Madhya Pradesh	45.0	20.704444	0.835289	18.90	20.1000	20.700	21.2000	22.60
Odisha	30.0	21.939333	0.998088	20.28	21.1600	21.715	22.8100	24.20
Rajasthan	32.0	20.125000	0.855344	17.40	19.6750	20.100	20.7250	21.90
Uttar Pradesh	70.0	21.442429	0.763448	19.39	20.9450	21.475	21.9700	23.10
Uttarakhand	13.0	21.995385	0.826787	20.90	21.6300	21.880	22.4000	23.71

Conclusion: The result is showing the count, mean, standard deviation, min, max, and percentages of the columns.

Question 2: 2. Which State in India had the most people suffering from fever per 100,000 people?

In [20]:
```
#Pivot table showing the total mean of people suffering from fever per 100,000 pe
#will display the states with the top means per 100,000 people
Health13=pd.pivot_table(Health1,index='State Name',values='Fever100,000')
Health13
```

Out[20]:

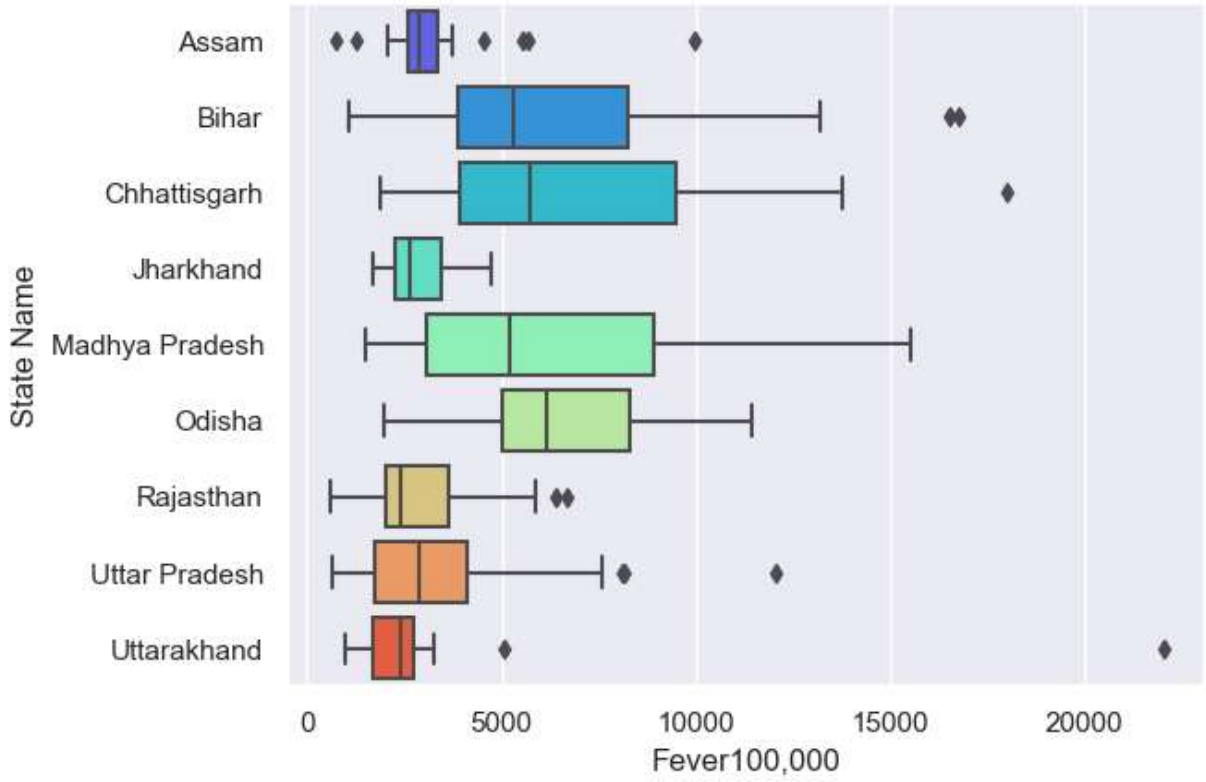
State Name	Fever100,000
Assam	3307.217391
Bihar	6602.162162
Chhattisgarh	7041.848750
Jharkhand	2891.666667
Madhya Pradesh	6134.139333
Odisha	6512.720667
Rajasthan	2944.843750
Uttar Pradesh	3338.687286
Uttarakhand	3861.423846

Conclusion: The pivot table shows that Chhattisgarh is the top city with the highest mean at 7042 people sick per 100,000 people. 7042 is 7.42% of 100,000 people for the city of Chhattisgarh.

Plot of Pivot Table Results

In [21]:
```
#boxplot displays the the distibution of quantitative data that shows comparisons
#The box shows the quartiles of the dataset while the whiskers extend to show the
sns.boxplot(x="Fever100,000", y="State Name", data=Health1,palette='rainbow')
```

Out[21]:
```
<AxesSubplot:xlabel='Fever100,000', ylabel='State Name'>
```



Conclusion: The boxplot is showing the result of the pivot table and concluding that Chhattisgarh is has the highest mean at 7042 people, 7.42% of the 100,000 people.

Groupby() and Describe() for analysis

In [22]:
```
#Display State Name and Fever100,000 for descriptive results on columns
Health20=Health1[['State Name','Fever100,000']]
Health20.groupby('State Name').describe()
```

Out[22]:

	Fever100,000							
	count	mean	std	min	25%	50%	75%	max
State Name								
Assam	23.0	3307.217391	1837.543443	748.00	2599.5000	2892.000	3340.0000	9987.00
Bihar	37.0	6602.162162	3901.674480	1085.00	3885.0000	5288.000	8236.0000	16778.00
Chhattisgarh	16.0	7041.848750	4395.985064	1902.82	3947.7825	5748.205	9500.3200	18010.40
Jharkhand	18.0	2891.666667	929.454299	1688.00	2254.2500	2656.000	3444.5000	4719.00
Madhya Pradesh	45.0	6134.139333	3809.256452	1498.41	3070.8500	5217.760	8907.6500	15542.68
Odisha	30.0	6512.720667	2499.319145	1983.42	5028.2575	6145.555	8312.8825	11461.72
Rajasthan	32.0	2944.843750	1524.767229	587.00	2007.5000	2409.500	3627.5000	6700.00
Uttar Pradesh	70.0	3338.687286	2174.126133	657.81	1749.3325	2890.890	4130.0350	12080.49
Uttarakhand	13.0	3861.423846	5557.080227	959.72	1667.9500	2418.480	2716.6600	22034.12

Conclusion: The result is showing the count, mean, standard deviation, min, max, and percentages of the columns.

Question 3: 3. Which State in India had the most females suffering from Tuberculosis per 100,000 people?

In [23]:
```
#Pivot table showing which state in India had the most females suffering from Tub
#will display the states with the highest mean per 100,000 people
Health14=pd.pivot_table(Health1,index='State Name',values='FemaleTuber100,000')
Health14
```

Out[23]:

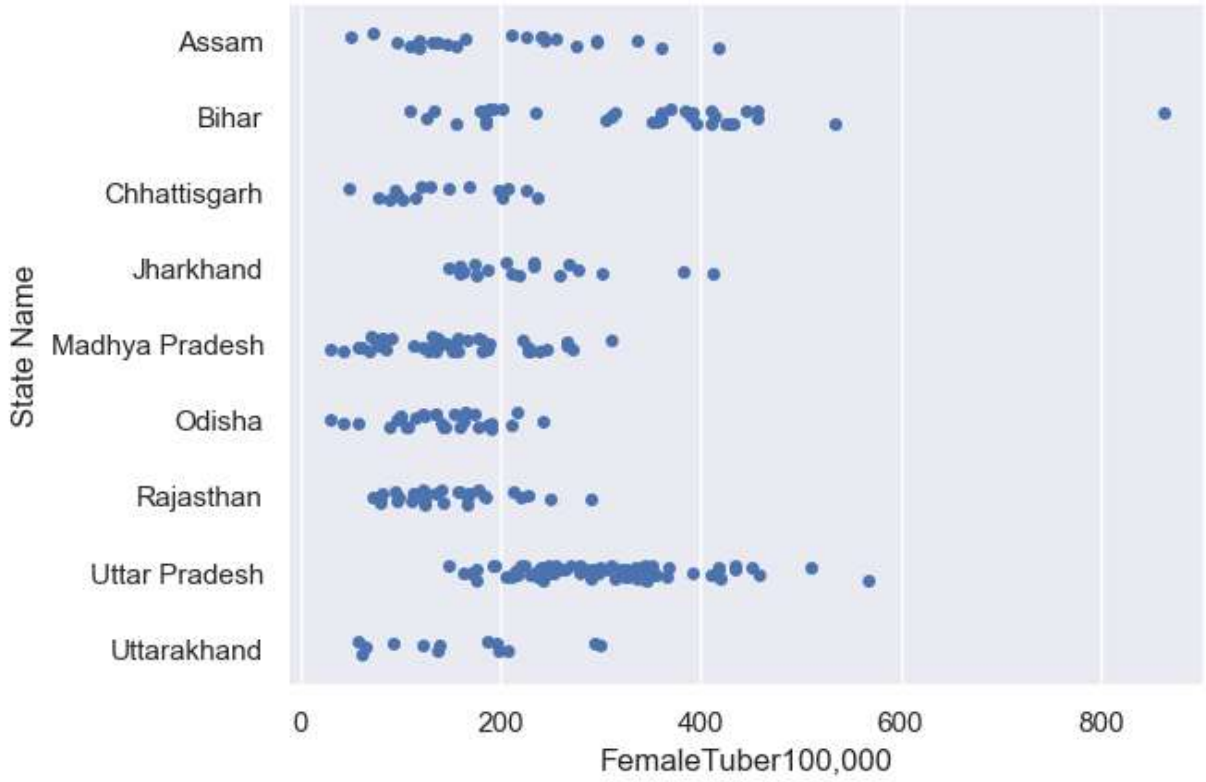
State Name	FemaleTuber100,000
Assam	204.000000
Bihar	332.783784
Chhattisgarh	140.593750
Jharkhand	231.222222
Madhya Pradesh	151.203556
Odisha	139.140667
Rajasthan	146.968750
Uttar Pradesh	304.687000
Uttarakhand	157.899231

Conclusion: The pivot table shows that Bihar had 333 females suffering from Tuberculosis per 100,000 people. 333 is 0.00333% of the 100,000 people.

Plot of Pivot Table Results

In [24]:
```
#strip plot showing the categorical vs the numeric
#scatter plot based on the State Name column category
sns.stripplot(x="FemaleTuber100,000", y="State Name", data=Health1)
```

Out[24]: <AxesSubplot:xlabel='FemaleTuber100,000', ylabel='State Name'>



Conclusion: The stripplot is showing the result of the pivot table and concluding that Bihar has the highest mean of people at 333, 0.00333% of 100,000 people.

Groupby() and Describe() for analysis

In [25]:
```
#Display State Name and FemaleTuber100,000 for descriptive results on columns
Health19=Health1[['State Name','FemaleTuber100,000']]
Health19.groupby('State Name').describe()
```

Out[25]:

	FemaleTuber100,000							
	count	mean	std	min	25%	50%	75%	max
State Name								
Assam	23.0	204.000000	97.888155	50.00	125.0000	210.000	265.000	417.00
Bihar	37.0	332.783784	144.669573	108.00	193.0000	360.000	411.000	863.00
Chhattisgarh	16.0	140.593750	58.067967	48.26	95.0425	123.970	198.065	235.65
Jharkhand	18.0	231.222222	75.571980	147.00	174.5000	214.000	265.500	412.00
Madhya Pradesh	45.0	151.203556	68.042760	28.99	90.1500	142.370	187.450	310.83
Odisha	30.0	139.140667	50.199407	30.16	106.2175	141.505	171.740	242.83
Rajasthan	32.0	146.968750	52.917150	71.00	112.5000	137.000	171.250	291.00
Uttar Pradesh	70.0	304.687000	85.109310	147.29	240.9000	303.620	345.995	566.96
Uttarakhand	13.0	157.899231	81.296642	57.74	92.9500	137.920	197.830	299.18

Conclusion: The result is showing the count, mean, standard deviation, min, max, and percentages of the columns.

Question 4: # 4. Which State in India had the highest total mean age of Males and females drop out of school between ages 6-17?

In [26]:
```
#pivot table showing which state in india had the highest mean age of males and f
#display the mean ages of states
Health15=pd.pivot_table(Health1,index='State Name',values='dropout6-17')
Health15
```

Out[26]:

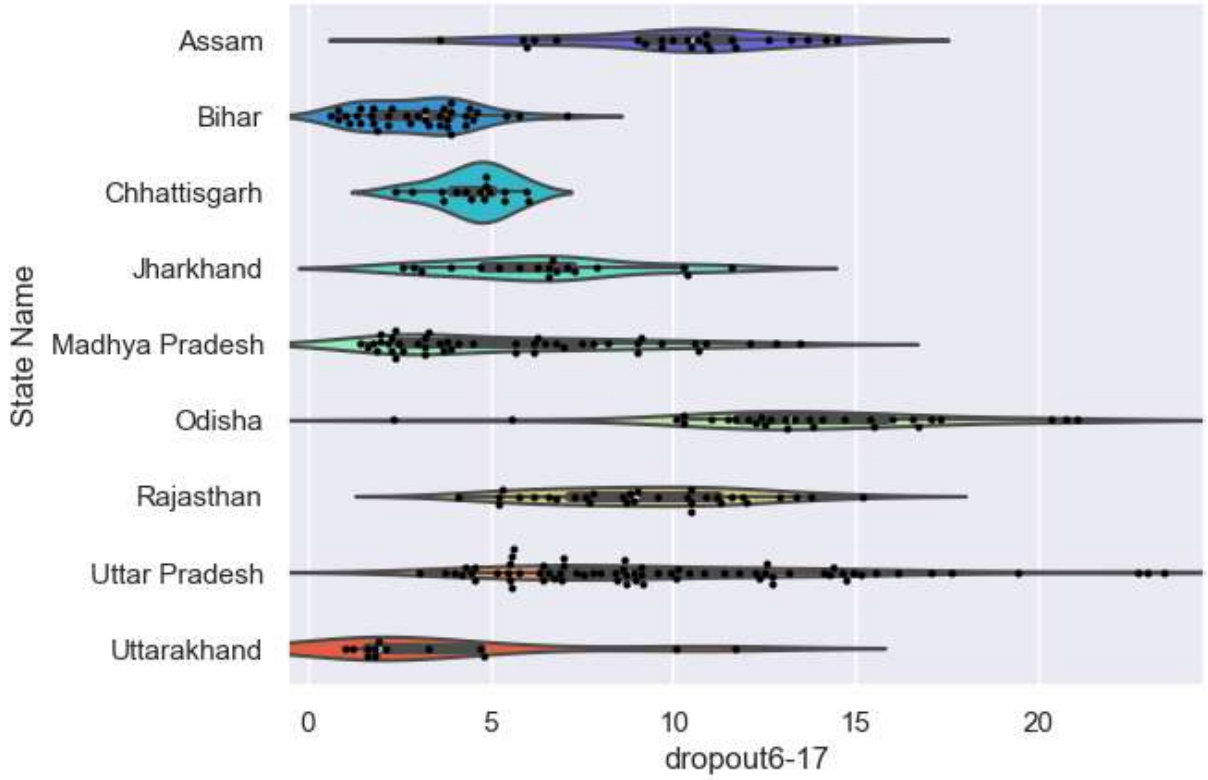
	dropout6-17
State Name	
Assam	10.086957
Bihar	3.002703
Chhattisgarh	4.520000
Jharkhand	6.433333
Madhya Pradesh	5.515556
Odisha	13.580333
Rajasthan	9.228125
Uttar Pradesh	9.924571
Uttarakhand	3.676923

Conclusion: The pivot table shows the highest mean age of school dropout ages between 6-17 are 13 and half years old from the state Odisha.

Plot of Pivot Table Results

In [27]:
```
# Combining Categorical Plots
#Used for Categorical Plots respresentation
sns.violinplot(x="dropout6-17", y="State Name", data=Health1,palette='rainbow')
sns.swarmplot(x="dropout6-17", y="State Name", data=Health1,color='black',size=3)
```

Out[27]:
```
<AxesSubplot:xlabel='dropout6-17', ylabel='State Name'>
```



Conclusion: The violinplot and swarmplot show the results of the pivot table concluding that Odisha has the highest mean age at 13 and a half years old.

Groupby() and Describe() for analysis

In [28]:
```
#Display State Name and dropout6-17 for descriptive results on columns
Health18=Health1[['State Name','dropout6-17']]
Health18.groupby('State Name').describe()
```

Out[28]:

	dropout6-17							
	count	mean	std	min	25%	50%	75%	max
State Name								
Assam	23.0	10.086957	2.841623	3.60	9.1000	10.500	11.6500	14.50
Bihar	37.0	3.002703	1.531065	0.60	1.8000	3.200	3.9000	7.10
Chhattisgarh	16.0	4.520000	1.022292	2.36	3.9500	4.760	5.0750	6.02
Jharkhand	18.0	6.433333	2.551124	2.60	4.8250	6.600	7.2500	11.60
Madhya Pradesh	45.0	5.515556	3.420390	1.40	2.5000	4.100	7.8000	13.50
Odisha	30.0	13.580333	3.984049	2.33	11.8075	13.230	15.8725	21.06
Rajasthan	32.0	9.228125	2.810721	4.10	7.1750	8.950	11.2250	15.20
Uttar Pradesh	70.0	9.924571	4.750595	3.03	6.4275	8.835	12.6625	23.44
Uttarakhand	13.0	3.676923	3.434235	1.01	1.6200	1.920	4.7100	11.69

Conclusion: The result is showing the count, mean, standard deviation, min, max, and percentages of the columns.

Question 5: 5. Which State in India had the most males sick from chronic illnesses per 100,000 people?

In [29]:
```
#Pivot table showing the state with the most males that were sick from chronic il
#Display the total of males per 100,000
Health16=pd.pivot_table(Health1,index='State Name',values='TotalMaleIll100,000')
Health16
```

Out[29]:

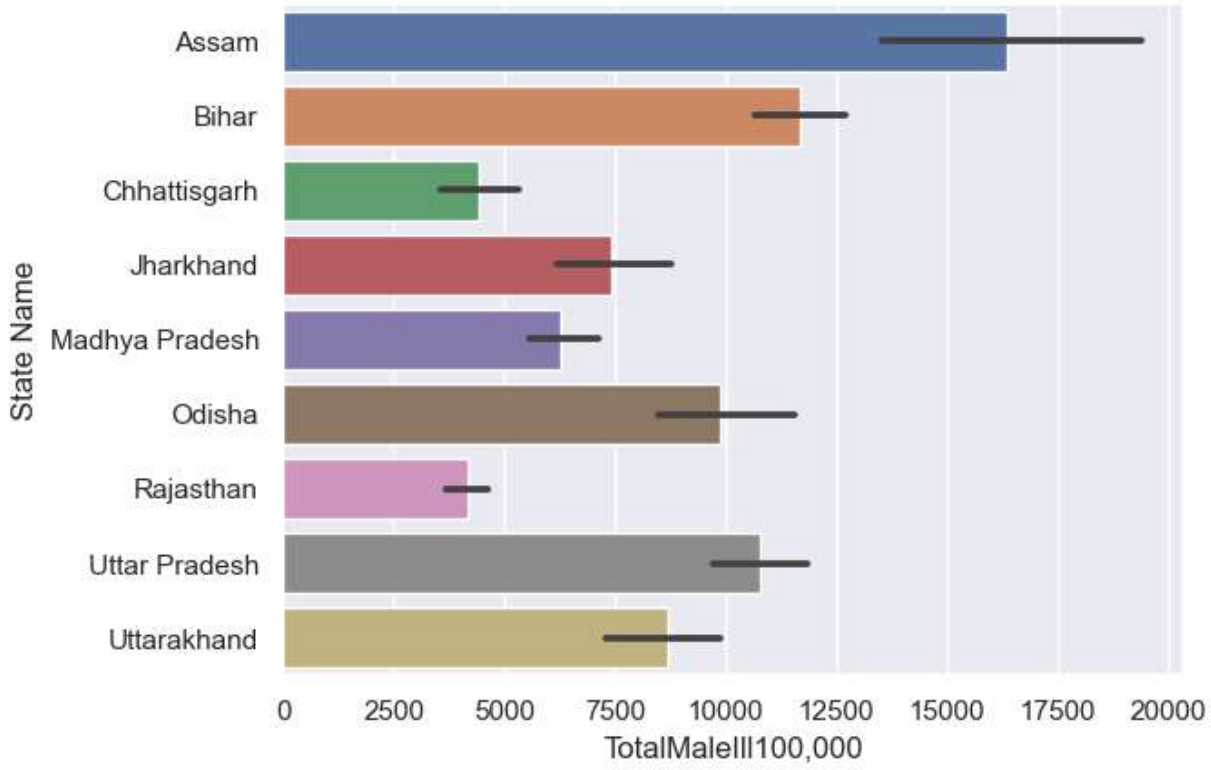
State Name	TotalMaleIll100,000
Assam	16360.478261
Bihar	11699.378378
Chhattisgarh	4401.625000
Jharkhand	7396.944444
Madhya Pradesh	6258.733333
Odisha	9891.900000
Rajasthan	4182.593750
Uttar Pradesh	10772.385714
Uttarakhand	8693.461538

Conclusion: The pivot table shows the highest total was in Assam with 16,360 people sick with chronic illness per 100,000 people. 16,360 is 16.36% of 100,000 people for the state if Assam.

Plot of Pivot Table Results

In [30]:
```
sns.barplot(x='TotalMaleIll100,000', y='State Name', data=Health1)
# a bar plot that shows the average or mean of TotalMaleIll100,000 per categorial
# average of Assam is higher for males
```

Out[30]: <AxesSubplot:xlabel='TotalMaleIll100,000', ylabel='State Name'>



Conclusion: The barplot shows the highest average of chronic illnesses is 16,360 people per 100,000. 16.360 is 16.36% of 100,000 for the State of Assam.

Groupby() and Describe() for analysis

In [31]:
```
#Display State Name and TotalMaleIll100,000 for descriptive results on columns
Health17=Health1[['State Name','TotalMaleIll100,000']]
Health17.groupby('State Name').describe()
```

Out[31]:

	TotalMaleIll100,000							
	count	mean	std	min	25%	50%	75%	max
State Name								
Assam	23.0	16360.478261	7685.027917	6557.0	10601.00	12438.0	23222.00	33411.0
Bihar	37.0	11699.378378	3253.986020	4599.0	10198.00	11697.0	12864.00	21972.0
Chhattisgarh	16.0	4401.625000	1846.760980	1897.0	3141.25	4090.0	5218.75	8604.0
Jharkhand	18.0	7396.944444	2702.082932	4762.0	5090.25	6554.5	8851.75	12972.0
Madhya Pradesh	45.0	6258.733333	2593.781106	2740.0	4508.00	5598.0	7617.00	16484.0
Odisha	30.0	9891.900000	4329.232076	3119.0	6575.00	10254.5	12607.25	21787.0
Rajasthan	32.0	4182.593750	1379.908713	1910.0	3022.00	4144.5	4885.25	7679.0
Uttar Pradesh	70.0	10772.385714	4441.694236	3676.0	7212.00	9194.0	15348.50	19207.0
Uttarakhand	13.0	8693.461538	2371.425050	4929.0	5970.00	9819.0	10594.00	11287.0

Conclusion: The result is showing the count, mean, standard deviation, min, max, and percentages of the columns.

Pearson Correlation Coefficient

Tests whether two samples have a linear relationship.Observations in each sample are independent and identically distributed (iid). Observations in each sample are normally distributed.

In [32]:
```
#Correlation test of the first row of data for dataset Health1
Health1[['MeanFemales','Total_Attend_6-17','dropout6-17','Fever100,000','TotalIll
```

Out[32]:

	MeanFemales	Total_Attend_6-17	dropout6-17	Fever100,000	TotalIll100,000	TotalMaleIll10
MeanFemales	1.0	-0.005583	0.168018	-0.016115	0.241112	0.24

Conclusion: We can see that the strongest correlation coefficient between meanfemales is with TotalMaleIll100,000.

Hypothesis Testing

Test if MeanFemales are normally distributed
Ho: Reject
H1: Fail to Reject

scipy.stats.mstats.normaltest

Tests whether a sample differs from a normal distribution.

This function tests the null hypothesis that a sample comes from a normal distribution.

In [34]:
```
#Display the first 5 rows
Health1.head()
```

Out[34]:

	State Name	MeanFemales	Total_Attend_6-17	dropout6-17	Fever100,000	TotalIll100,000	TotalMaleIll100,00
0	Assam	21.7	89.6	10.0	2721.0	11421.0	1080
1	Assam	21.7	90.7	9.2	2906.0	7497.0	719
2	Assam	22.6	85.5	13.2	9987.0	23339.0	2144
3	Assam	22.1	88.2	10.9	2054.0	10345.0	984
4	Assam	21.5	93.8	5.9	2704.0	7585.0	730

In [37]:
```
# this returns a test statistic and p value
# normaltest returns a 2-tuple of the chi-squared statistic,
# and the associated p-value.
normaltest(Health1['MeanFemales'])
```

Out[37]:
```
NormaltestResult(statistic=0.7461960401731753, pvalue=0.6885977331264207)
```

If P-Value is <0.5, Reject H0 null hypothesis, If P-Value is >0.5, Fail to reject H0 null hypothesis

Conclusion: The result is >0.5 and will fail to Reject H0

In [39]:
```
#Normal test 2
#Test Total_Attend_6-17 for P-Value
normaltest(Health1['Total_Attend_6-17'])
```

Out[39]:
```
NormaltestResult(statistic=27.07931883486887, pvalue=1.3176517044929738e-06)
```

If P-Value is <0.5, Reject H0 null hypothesis, If P-Value is >0.5, Fail to reject H0 null hypothesis

Conclusion: Reject H0 Null hypothesis for Total_Attend_6-17

In [40]:
```
#Normal test 3
#Test dropout6-17 for P-Value
normaltest(Health1['dropout6-17'])
```

Out[40]:
```
NormaltestResult(statistic=21.557248749066453, pvalue=2.0840249384308303e-05)
```

If P-Value is <0.5, Reject H0 null hypothesis, If P-Value is >0.5, Fail to reject H0 null hypothesis

Conclusion: Reject H0 Null hypothesis for dropout6-17

Conclusion

Questions and Answers

1. What State in India had the highest total mean marriages for females?

Answer: Assam was the top State in India with the total age for marriage for women at 22 years old.

2. Which State in India had the most people suffering from fever per 100,000 people?

Answer: Chhattisgarh is the top city with the highest mean at 7042 people sick per 100,000 people. 7042 is 7.42% of 100,000 people for the city of Chhattisgarh.

3. Which State in India had the most females suffering from Tuberculosis per 100,000 people?

Answer: Bihar had 333 females suffering from Tuberculosis per 100,000 people. 333 is 0.00333% of the 100,000 people.

4. Which State in India had the highest total mean age of Males and females drop out of school between ages 6-17?

Answer: the highest mean age of school dropout ages between 6-17 are 13 and half years old from the state Odisha.

5. Which State in India had the most males sick from chronic illnesses per 100,000 people?

Answer: the highest total was in Assam with 16,360 people sick with chronic illness per 100,000 people. 16,360 is 16.36% of 100,000 people for the state if Assam.