Group 4:

- Akshat (190030003-CSE)
- Ananya (190010011-CSE)
- Atul Singh (190020009-EE)
- Ashish (190030006-EE)
- Siddharth Patra (190030039-ME)
- Pruthvi Raj DJ (190030033-ME)

Data Analaysis Course Project :

Aim : To Collect, Analyze, Represent and Interpret the data for COIVD-19 from various data repositories.

INTRODUCTION

Coronavirus disease-19 (COVID-19) is a contagious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV2). The first case was identified in Wuhan, China in December 2019. Common symptoms of COVID-19 include fever, cough, fatigue, breathing difficulties, and loss of smell and taste. Symptoms begin one to fourteen days after exposure to the virus. While most people have mild symptoms, some people develop acute respiratory distress syndrome (ARDS). COVID-19 spreads via a number of means, primarily involving saliva and other bodily fluids and excretions. These fluids can form small droplets and aerosols, which can spread as an infected person breathes, coughs, sneezes, sings, or speaks. This Data Analysis project on Covid The project is a course project at Indian Institute of Technology, Dharwad which uses the various techniques of data analysis covered in the course. It consists of mainly 3 sections and is built completely over jupyter notebook using Python 3:

As a part of this course project for Data Analysis course our aim was to: - Collect
- Analyze
- Represent
- Interpret

the data for COVID-19 from various data repositories.

In this project we hypothesise about the effects of COVID-19 and its repercussions. We then use the collected data to negate or support the hypothesis. We use python to plot immaculate graphs, help us analyse, interpret and draw conclusions from the data. Due to the massive amounts of data that we have collected using a code makes it much easier to work with.

Importing python libraries to be used:

In [1]:
```
# Import necessary libraries
import numpy as np
import pandas as pd
from openpyxl import load_workbook
from datetime import date
%matplotlib inline
import matplotlib as mpl
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import seaborn as sns
import os # to work with local directory
import re
import plotly.express as px
import scipy.stats
from scipy.stats import pearsonr
from scipy.interpolate import interp1d
from itertools import islice
import seaborn as sb
import math
import statsmodels.formula.api as smf
import statsmodels.stats.multicomp as multi
import statsmodels.api as sm
```

Collection of Data

Collection of data was the first step to execute this project. In order to collect data we used the internet as our primary source for accurate data collection and real time information. We combed through numerous websites to get the most precise, congruous and apt statistics. The URLs of these websites have been added below for our future reference and also for the reference of others.

https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases for total number of COVID cases worldwide .
https://www.statista.com/statistics/1110522/india-number-of-coronavirus-cases-by-age-group/ for age group wise data.
https://www.researchgate.net/publication/340050986_The_Impact_of_Cross-Cultural_Differences_in_Handwashing_Patterns_on_the_COVID-19_Outbreak_Magnitude for handwashing data of different countries.
https://data.humdata.org/dataset/novel-coronavirus-2019-ncov-cases for total number of COVID deaths worldwide .
https://www.ecdc.europa.eu/en/publications-data/download-todays-data-geographic-distribution-covid-19-cases-worldwide for total case density worldwide.
https://pib.gov.in/PressReleasePage.aspx?PRID=1638804 for trade statistics of India.
https://en.wikipedia.org/wiki/List_of_countries_by_average_elevation for average altitude of each country.
https://ourworldindata.org/grapher/deaths-covid-19-vs-case-fatality-rate?tab=table&time=2020-11-27 for Total cases,deaths and several other covid related factors per country.
https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_by_population_density for population density of each country.
https://tradingeconomics.com/india/gdp-from-utilities for the indian economy statistics.

We shall now import the selected data to be used in the project using the openpyxl and panda libraries in python:

Importing Data

Deriving data from local files

Each file is stored in a dataframe (variable with names starting with df) of pandas and is then accessed from within the project

In [2]:

```
# To convert openpyxl files to data frames
def toDataFrame(ws):
    data = ws.values
    cols = next(data)[1:]
    data = list(data)
    idx = [r[0] for r in data]
    data = (islice(r, 1, None) for r in data)
    return pd.DataFrame(data, index=idx, columns=cols)
```

In [3]:

```
wb_JHU_confirmed = load_workbook(r"OrigData\\time-series-covid19-confirmed-global.xlsx")
ws_JHU_confirmed = wb_JHU_confirmed[wb_JHU_confirmed.sheetnames[0]]
df_JHU_confirmed = toDataFrame(ws_JHU_confirmed)

wb_JHU_fatality = load_workbook(r"OrigData\\time_series_covid19_deaths_global.xlsx")
ws_JHU_fatality = wb_JHU_fatality[wb_JHU_fatality.sheetnames[0]]
df_JHU_fatality = toDataFrame(ws_JHU_fatality)

wb_covid_data = load_workbook(r"OrigData\\owid-covid-data.xlsx")
ws_covid_data = wb_covid_data[wb_covid_data.sheetnames[0]]
df_covid_data = toDataFrame(ws_covid_data)

wb_age_data = load_workbook(r"OrigData\\statistic_id1110522_number-of-covid-19-cases-india-2020-by-age-group.xl
ws_age_data = wb_age_data[wb_age_data.sheetnames[1]]
df_age_data = pd.DataFrame(ws_age_data.values)

wb_altitude = load_workbook(r"OrigData\\DA data altitude.xlsx")
ws_altitude = wb_altitude[wb_altitude.sheetnames[0]]
df_altitude = toDataFrame(ws_altitude)

wb_economy = load_workbook(r"OrigData\\GDP20192020.xlsx")  # Work Book
ws_economy = wb_economy[wb_economy.sheetnames[0]]
df_economy = toDataFrame(ws_economy)

wb_economy_IND = load_workbook(r"OrigData\\IND_eco.xlsx")
ws_economy_IND = wb_economy_IND[wb_economy_IND.sheetnames[0]]
df_economy_IND = toDataFrame(ws_economy_IND)

wb_IND_inf = load_workbook(r"OrigData\\Inflation2019And2020.xlsx")
```

```
ws_IND_inf = wb_IND_inf[wb_IND_inf.sheetnames[0]]
df_IND_inf = toDataFrame(ws_IND_inf)
```

Now that we have stored the majority of the data in dataframes, we shall now display a **random sample** of data in each data frame to display the various columns and depict the data at hand

Assessing the Imported Data Visually

Here, we have displayed the data in its raw form that we are using in the project, visually

In [4]:
```
df_JHU_fatality.sample(n=5)
```

Out[4]:

	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	...	11/21/
Australian Capital Territory	Australia	-35.47350	149.012400	0	0	0	0	0	0	0	...	
NaN	Slovenia	46.15120	14.995500	0	0	0	0	0	0	0	...	1(
NaN	Kyrgyzstan	41.20438	74.766098	0	0	0	0	0	0	0	...	1:
Bonaire, Sint Eustatius and Saba	Netherlands	12.17840	-68.238500	0	0	0	0	0	0	0	...	
NaN	Bangladesh	23.68500	90.356300	0	0	0	0	0	0	0	...	6:

5 rows × 317 columns

In [5]:
```
df_JHU_confirmed.head(n=5)
```

Out[5]:

	Country/Region	Lat	Long	1/22/20	1/23/20	1/24/20	1/25/20	1/26/20	1/27/20	1/28/20	...	11/21/20	1
NaN	Afghanistan	33.93911	67.709953	0	0	0	0	0	0	0	...	44503	
NaN	Albania	41.15330	20.168300	0	0	0	0	0	0	0	...	32196	
NaN	Algeria	28.03390	1.659600	0	0	0	0	0	0	0	...	73774	
NaN	Andorra	42.50630	1.521800	0	0	0	0	0	0	0	...	6207	
NaN	Angola	-11.20270	17.873900	0	0	0	0	0	0	0	...	14413	

5 rows × 317 columns

In [6]:
```
df_country_cases = df_JHU_confirmed[['Country/Region','Lat','Long','11/30/20']]
df_country_cases.sample(5)
```

Out[6]:

	Country/Region	Lat	Long	11/30/20
Queensland	Australia	-27.469800	153.025100	1202
Western Australia	Australia	-31.950500	115.860500	821
Hainan	China	19.195900	109.745300	171
NaN	Saudi Arabia	23.885942	45.079162	357360
NaN	Uruguay	-32.522800	-55.765800	5857

In [7]:
```
df_covid_data.head(n=5)
```

Out[7]:

	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothe
AFG	Asia	Afghanistan	2020-01-23	NaN	0.0	NaN	NaN	0.0	Na
AFG	Asia	Afghanistan	2020-01-24	NaN	0.0	NaN	NaN	0.0	Na
AFG	Asia	Afghanistan	2020-01-25	NaN	0.0	NaN	NaN	0.0	Na
AFG	Asia	Afghanistan	2020-01-26	NaN	0.0	NaN	NaN	0.0	Na

	continent	location	date	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothe
AFG	Asia	Afghanistan	2020-01-27	NaN	0.0	NaN	NaN	0.0	Na

5 rows × 49 columns

In [8]:
```
df_age_data.iloc[[5,6, 7, 8,9,10], [1, 2]]
```

Out[8]:

	1	2
5	Less than 14 years	0.5
6	15-29 years	2.5
7	30-44 years	11.4
8	45-59 years	35.1
9	60-74 years	40.2
10	More than 75 years	10.3

In [9]:
```
df_altitude.head(5)
```

Out[9]:

	Fatality ratio	No of cases	Altitude	NaN	population density	NaN
Afghanistan	0.0383	1737.0	1,885 m (6,184 ft)	Afghanistan	49.0	None
Algeria	0.0297	2352.0	800 m (2,625 ft)	Algeria	18.0	None
Angola	0.0229	340.0	1,112 m (3,648 ft)	Angola	23.0	None
Argentina	0.0271	37941.0	595 m (1,952 ft)	Argentina	16.0	None
Australia	0.0325	907.0	330 m (1,083 ft)	Australia	3.0	None

In [10]:
```
df_economy_IND.head(5)
```

Out[10]:

	INFLATION RATE	INDIA INFRASTRUCTURE OUTPUT	LENDING RATES	REPO RATES	GOVERNMENT REVENUES	CONSUMER SPENDING	NaN	QUARTER	Business Expectations Index (BEI)
2019-11-01	5.54	0.7	9.4	4.9	10122.2	19119	None	2018 Q1	115
2019-12-01	7.35	3.1	9.4	4.9	11779.2	19841.8	None	2018Q2	115.8
2020-01-01	7.59	2.2	9.4	4.9	12828.6	18890.1	None	2018 Q3	114.6
2020-02-01	6.58	6.4	9.4	4.9	14288.7	19038.5	None	2018 Q4	114.1
2020-03-01	5.84	-8.6	9.4	4	17507.3	20464.2	None	2019 Q1	115

In [11]:
```
df_IND_inf_modif = df_IND_inf[:-4]
df_IND_inf_modif = df_IND_inf_modif.loc[:,'2015':'2023']
df_IND_inf_modif
```

Out[11]:

	2015	2016	2017	2018	2019	2020	2021	2022	2023
Australia	1.5	1.3	2.0	1.9	1.6	0.7	1.3	1.5	1.9
Canada	1.1	1.4	1.6	2.3	1.9	0.6	1.3	1.6	1.9
China, People's Republic of	1.4	2.0	1.6	2.1	2.9	2.9	2.7	2.6	2.6
Ethiopia	9.6	6.6	10.7	13.8	15.8	20.2	11.5	8.0	8
France	0.1	0.3	1.2	2.1	1.3	0.5	0.6	1.0	1.2
Germany	0.7	0.4	1.7	2.0	1.3	0.5	1.1	1.3	1.5
India	4.9	4.5	3.6	3.4	4.8	4.9	3.7	3.8	3.9
Iraq	1.4	0.5	0.1	0.4	-0.2	0.8	1.0	1.5	1.8

	2015	2016	2017	2018	2019	2020	2021	2022	2023
Japan	0.8	-0.1	0.5	1.0	0.5	-0.1	0.3	0.7	0.8
Kenya	6.6	6.3	8.0	4.7	5.2	5.3	5.0	5.0	5
United States	0.1	1.3	2.1	2.4	1.8	1.5	2.8	2.1	2.1

Organising the Imported Data via Program

In this category, we have modified the raw data into the format we will be using in the project

We begin by printing the names of the countries available in the data set of the two sources namely:

1. John Hopkins Dataset(JHD)
2. Our World In Data(OWID)

The list of countries available in John Hopkins University data set are:

In [12]:
```
# List of countries available in JHU and OWID data sets
print("countries available in John Hopkins Dataset :")
df_JHU_confirmed['Country/Region'].unique()
```

```
countries available in John Hopkins Dataset :
```

Out[12]:
```
array(['Afghanistan', 'Albania', 'Algeria', 'Andorra', 'Angola',
       'Antigua and Barbuda', 'Argentina', 'Armenia', 'Australia',
       'Austria', 'Azerbaijan', 'Bahamas', 'Bahrain', 'Bangladesh',
       'Barbados', 'Belarus', 'Belgium', 'Belize', 'Benin', 'Bhutan',
       'Bolivia', 'Bosnia and Herzegovina', 'Botswana', 'Brazil',
       'Brunei', 'Bulgaria', 'Burkina Faso', 'Burma', 'Burundi',
       'Cabo Verde', 'Cambodia', 'Cameroon', 'Canada',
       'Central African Republic', 'Chad', 'Chile', 'China', 'Colombia',
       'Comoros', 'Congo (Brazzaville)', 'Congo (Kinshasa)', 'Costa Rica',
       "Cote d'Ivoire", 'Croatia', 'Cuba', 'Cyprus', 'Czechia', 'Denmark',
       'Diamond Princess', 'Djibouti', 'Dominica', 'Dominican Republic',
       'Ecuador', 'Egypt', 'El Salvador', 'Equatorial Guinea', 'Eritrea',
       'Estonia', 'Eswatini', 'Ethiopia', 'Fiji', 'Finland', 'France',
       'Gabon', 'Gambia', 'Georgia', 'Germany', 'Ghana', 'Greece',
       'Grenada', 'Guatemala', 'Guinea', 'Guinea-Bissau', 'Guyana',
       'Haiti', 'Holy See', 'Honduras', 'Hungary', 'Iceland', 'India',
       'Indonesia', 'Iran', 'Iraq', 'Ireland', 'Israel', 'Italy',
       'Jamaica', 'Japan', 'Jordan', 'Kazakhstan', 'Kenya',
       'Korea, South', 'Kosovo', 'Kuwait', 'Kyrgyzstan', 'Laos', 'Latvia',
       'Lebanon', 'Lesotho', 'Liberia', 'Libya', 'Liechtenstein',
       'Lithuania', 'Luxembourg', 'MS Zaandam', 'Madagascar', 'Malawi',
       'Malaysia', 'Maldives', 'Mali', 'Malta', 'Marshall Islands',
       'Mauritania', 'Mauritius', 'Mexico', 'Moldova', 'Monaco',
       'Mongolia', 'Montenegro', 'Morocco', 'Mozambique', 'Namibia',
       'Nepal', 'Netherlands', 'New Zealand', 'Nicaragua', 'Niger',
       'Nigeria', 'North Macedonia', 'Norway', 'Oman', 'Pakistan',
       'Panama', 'Papua New Guinea', 'Paraguay', 'Peru', 'Philippines',
       'Poland', 'Portugal', 'Qatar', 'Romania', 'Russia', 'Rwanda',
       'Saint Kitts and Nevis', 'Saint Lucia',
       'Saint Vincent and the Grenadines', 'San Marino',
       'Sao Tome and Principe', 'Saudi Arabia', 'Senegal', 'Serbia',
       'Seychelles', 'Sierra Leone', 'Singapore', 'Slovakia', 'Slovenia',
       'Solomon Islands', 'Somalia', 'South Africa', 'South Sudan',
       'Spain', 'Sri Lanka', 'Sudan', 'Suriname', 'Sweden', 'Switzerland',
       'Syria', 'Taiwan*', 'Tajikistan', 'Tanzania', 'Thailand',
       'Timor-Leste', 'Togo', 'Trinidad and Tobago', 'Tunisia', 'Turkey',
       'US', 'Uganda', 'Ukraine', 'United Arab Emirates',
       'United Kingdom', 'Uruguay', 'Uzbekistan', 'Vanuatu', 'Venezuela',
       'Vietnam', 'West Bank and Gaza', 'Western Sahara', 'Yemen',
       'Zambia', 'Zimbabwe'], dtype=object)
```

The countries available in OWID data set are :

In [13]:
```
print("countries available in OWID :")
df_covid_data['location'].unique()
```

```
countries available in OWID :
```

Out[13]:
```
array(['Afghanistan', 'Angola', 'Albania', 'Andorra',
       'United Arab Emirates', 'Argentina', 'Armenia',
       'Antigua and Barbuda', 'Australia', 'Austria', 'Azerbaijan',
       'Burundi', 'Belgium', 'Benin', 'Burkina Faso', 'Bangladesh',
       'Bulgaria', 'Bahrain', 'Bahamas', 'Bosnia and Herzegovina',
       'Belarus', 'Belize', 'Bolivia', 'Brazil', 'Barbados', 'Brunei',
       'Bhutan', 'Botswana', 'Central African Republic', 'Canada',
       'Switzerland', 'Chile', 'China', "Cote d'Ivoire", 'Cameroon',
       'Democratic Republic of Congo', 'Congo', 'Colombia', 'Comoros',
       'Cape Verde', 'Costa Rica', 'Cuba', 'Cyprus', 'Czech Republic',
       'Germany', 'Djibouti', 'Dominica', 'Denmark', 'Dominican Republic',
       'Algeria', 'Ecuador', 'Egypt', 'Eritrea', 'Western Sahara',
```

```
       'Spain', 'Estonia', 'Ethiopia', 'Finland', 'Fiji', 'France',
       'Gabon', 'United Kingdom', 'Georgia', 'Ghana', 'Guinea', 'Gambia',
       'Guinea-Bissau', 'Equatorial Guinea', 'Greece', 'Grenada',
       'Guatemala', 'Guyana', 'Hong Kong', 'Honduras', 'Croatia', 'Haiti',
       'Hungary', 'Indonesia', 'India', 'Ireland', 'Iran', 'Iraq',
       'Iceland', 'Israel', 'Italy', 'Jamaica', 'Jordan', 'Japan',
       'Kazakhstan', 'Kenya', 'Kyrgyzstan', 'Cambodia',
       'Saint Kitts and Nevis', 'South Korea', 'Kuwait', 'Laos',
       'Lebanon', 'Liberia', 'Libya', 'Saint Lucia', 'Liechtenstein',
       'Sri Lanka', 'Lesotho', 'Lithuania', 'Luxembourg', 'Latvia',
       'Morocco', 'Monaco', 'Moldova', 'Madagascar', 'Maldives', 'Mexico',
       'Marshall Islands', 'Macedonia', 'Mali', 'Malta', 'Myanmar',
       'Montenegro', 'Mongolia', 'Mozambique', 'Mauritania', 'Mauritius',
       'Malawi', 'Malaysia', 'Namibia', 'Niger', 'Nigeria', 'Nicaragua',
       'Netherlands', 'Norway', 'Nepal', 'New Zealand', 'Oman',
       'Pakistan', 'Panama', 'Peru', 'Philippines', 'Papua New Guinea',
       'Poland', 'Portugal', 'Paraguay', 'Palestine', 'Qatar', 'Romania',
       'Russia', 'Rwanda', 'Saudi Arabia', 'Sudan', 'Senegal',
       'Singapore', 'Solomon Islands', 'Sierra Leone', 'El Salvador',
       'San Marino', 'Somalia', 'Serbia', 'South Sudan',
       'Sao Tome and Principe', 'Suriname', 'Slovakia', 'Slovenia',
       'Sweden', 'Swaziland', 'Seychelles', 'Syria', 'Chad', 'Togo',
       'Thailand', 'Tajikistan', 'Timor', 'Trinidad and Tobago',
       'Tunisia', 'Turkey', 'Taiwan', 'Tanzania', 'Uganda', 'Ukraine',
       'Uruguay', 'United States', 'Uzbekistan', 'Vatican',
       'Saint Vincent and the Grenadines', 'Venezuela', 'Vietnam',
       'Vanuatu', 'Kosovo', 'Yemen', 'South Africa', 'Zambia', 'Zimbabwe',
       'World', 'International'], dtype=object)
```

We shall now organise the data into dataframes and/or data structures like that of dictionaries in python

In [14]:

```
letters = "a b c d e f g h i j k l m n o p q r s t u v w x y z".upper().split()
alphas = []
for i in letters:
    alphas.append(i)
for i in letters:
    for j in letters:
        k = (i+j)
        alphas.append(k)
        if k == 'LF':
            break
    if i == 'L':
        break
alphas = alphas[4:]
```

Collecting total cases in world using JHU data:

In [15]:

```
totalCasesInWorldJHU = []
dateTotalCasesInWorldJHU = []
for X in alphas:
    column = ws_JHU_confirmed[X]
    Sum = 0
    for i in range(1,272):
        Sum+=column[i].value
    if isinstance(column[0].value, str):
        dateTotalCasesInWorldJHU.append(column[0].value)
    else:
        dateTotalCasesInWorldJHU.append(column[0].value.strftime('%d/%m/%Y'))
    totalCasesInWorldJHU.append(Sum)
```

Collecting total deaths in world using JHU data:

In [16]:

```
totalDeathsInWorldJHU = []
dateTotalDeathsInWorldJHU = []
for X in alphas:
    column = ws_JHU_fatality[X]
    Sum = 0
    for i in range(1,272):
        Sum+=column[i].value
    if isinstance(column[0].value, str):
        dateTotalDeathsInWorldJHU.append(column[0].value)
    else:
        dateTotalDeathsInWorldJHU.append(column[0].value.strftime('%d/%m/%Y'))
    totalDeathsInWorldJHU.append(Sum)
```

Storing all data from OWID into a dictionary

In [17]:

```
dataCol = "a b c d e f g h i j k l m n o p q r s t u v w x y z aa ab ac ad ae af ag ah ai aj ak al am an ao ap
dataOWID = {}
for X in dataCol:
    column = ws_covid_data[X]
    sample = []
    for x in range(1,len(column)):
```

```
            sample.append(column[x].value)
    dataOWID[column[0].value] = sample
def dailyKeyReturnerUsingLoc(code,key):
    loc = dataOWID['location']
    date = dataOWID['date']
    total_cases = dataOWID[key]
    total_cases_IND = []
    date_IND = []
    for x in range(len(loc)):
        if loc[x] == code:
            total_cases_IND.append(total_cases[x])
            date_IND.append(date[x])
    return [date_IND,total_cases_IND]
# print(dataOWID['date'])
```

In [18]:

```
dataCol = "b c d e f g".upper().split(" ")
dataEconInd = {}
for X in dataCol:
    column = ws_economy_IND[X]
    sample = []
    for x in range(3,13):
        sample.append(column[x].value)
    dataEconInd[column[0].value] = sample
# dataEconInd
def economyGetter(key):
    return dataEconInd[key]
```

In [19]:

```
dataCol = "a b c e f g h i j k l m n o p q r s t u v w x y z aa ab ac ad ae af ag ah ai aj ak al am an ao ap aq
dataOWID_today = {}
for X in dataCol:
    column = ws_covid_data[X]
    sample = []
    dates = ws_covid_data['D']
    for x in range(1,len(column)):
        if dates[x].value == '2020-11-29':
            sample.append(column[x].value)
    dataOWID_today[column[0].value] = sample
# dataOWID_today['location']
```

We shall now begin formatting the data at hand. We have displayed the data after formatting in each set

Creating a dataframe that stores the current data of the countries present in OWID data set

In [20]:

```
df_cases_today = pd.DataFrame.from_dict(dataOWID_today)
df_cases_today.sample(n=5)
```

Out[20]:

	iso_code	continent	location	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoo
121	MWI	Africa	Malawi	6025	0	3.143	185.0	0	
160	SWE	Europe	Sweden	243129	0	4976.286	6681.0	0	3
66	GNB	Africa	Guinea-Bissau	2422	0	0.143	43.0	0	
28	CAF	Africa	Central African Republic	4913	0	0.286	63.0	0	
19	BIH	Europe	Bosnia and Herzegovina	87374	664	1052.571	2620.0	44	4

5 rows × 49 columns

Creating a dataframe that stores the current data from OWID data set from selected countries

In [21]:

```
df_cases_selected_countries = df_cases_today.query('location in ["India","Italy","United States","Japan"]')
df_cases_selected_countries
```

Out[21]:

	iso_code	continent	location	total_cases	new_cases	new_cases_smoothed	total_deaths	new_deaths	new_deaths_smoothe
77	IND	Asia	India	9431691	38772	41689.429	137139.0	443	485.85
83	ITA	Europe	Italy	1585178	20646	25187.143	54904.0	541	725.85
86	JPN	Asia	Japan	147515	2058	2068.714	2057.0	15	16.28
177	USA	North America	United States	13383320	138903	162364.857	266873.0	826	1429.57

4 rows × 49 columns

Creating a dataframe where all the attributes related to diseases are not null/None from the current status dataframe derived originally from the OWID dataframe.

In [22]:
```
disease_list =  ['cardiovasc_death_rate','female_smokers','male_smokers','diabetes_prevalence','total_deaths_pe
df_disease = df_cases_today.dropna(subset=disease_list)
colsList = ['location','total_deaths_per_million','cardiovasc_death_rate','female_smokers','male_smokers','diab
df_disease[colsList].sample(n=5)
```

Out[22]:

	location	total_deaths_per_million	cardiovasc_death_rate	female_smokers	male_smokers	diabetes_prevalence
105	Morocco	156.839	419.146	0.8	47.1	7.14
174	Uganda	4.394	213.333	3.4	16.7	2.50
61	United Kingdom	859.411	122.137	20.0	24.7	4.28
77	India	99.376	282.280	1.9	20.6	10.39
171	Turkey	160.756	171.285	14.1	41.1	12.13

Creating a dataframe in a similar way as above but for age groups data this time

In [23]:
```
ageGroupList = ['total_cases_per_million','aged_70_older','aged_65_older','median_age']
df_ageGroupData = df_cases_today.dropna(subset=ageGroupList)
colsList = ['location','total_cases_per_million','aged_70_older','aged_65_older','median_age']
df_ageGroupData[colsList].sample(n=5)
```

Out[23]:

	location	total_cases_per_million	aged_70_older	aged_65_older	median_age
9	Austria	31056.582	13.748	19.202	44.4
147	Senegal	960.049	1.796	3.008	18.7
19	Bosnia and Herzegovina	26631.797	10.711	16.569	42.5
161	Swaziland	5525.081	1.845	3.163	21.5
158	Slovakia	19366.285	9.167	15.070	41.2

Duplicating the altitude data dataframe to have integral values for altitude (in m) for easier analysis

In [24]:
```
altitudeList = ['Altitude','Fatality ratio','No of cases']
alt = df_altitude['Altitude']
df_altitude_modif = df_altitude.copy()
for i in range(len(alt)):
    if alt[i]!=None:
        temp = alt[i]
        temp = temp.split('m')
        temp = temp[0]
        temp = float(temp.replace(',','').replace(' ',''))
        df_altitude_modif.iloc[i, df_altitude_modif.columns.get_loc('Altitude')] = temp
df_altitude_modif.sample(n=5)
```

Out[24]:

	Fatality ratio	No of cases	Altitude	NaN	population density	NaN
Ghana	0.0063	323.0	190	Ghana	127.0	None
Portugal	0.0150	4209.0	372	Portugal	112.0	None
France	0.0233	50957.0	375	France	123.0	None
Zimbabwe	0.0285	274.0	961	Zimbabwe	39.0	None
Zambia	0.0203	357.0	1138	Zambia	22.0	None

Creating a dataframe where all the attributes related to economy are not null/None from the current status dataframe derived originally from the OWID dataframe.

In [25]:
```
economy_list =  ['gdp_per_capita','human_development_index','extreme_poverty','total_deaths_per_million','total
df_economy = df_cases_today.dropna(subset=disease_list)
colsList = ['location','total_deaths_per_million','gdp_per_capita','extreme_poverty','human_development_index',
df_disease[colsList].sample(n=5)
```

Out[25]:

	location	total_deaths_per_million	gdp_per_capita	extreme_poverty	human_development_index	total_cases_per_million
24	Barbados	24.359	16978.068	NaN	0.800	956.951

	location	total_deaths_per_million	gdp_per_capita	extreme_poverty	human_development_index	total_cases_per_million
38	Comoros	8.050	1413.890	18.1	0.503	702.626
133	Panama	709.192	22267.037	2.2	0.789	38177.951
30	Switzerland	537.285	57410.166	NaN	0.944	36776.898
45	Djibouti	61.741	2705.406	22.5	0.476	5745.940

Creating a dataframe where all the attributes related to medical facilities across various countries are not null/None from the current status dataframe derived originally from the OWID dataframe.

In [26]:
```
medical_list =  ['life_expectancy','handwashing_facilities','hospital_beds_per_thousand','total_deaths_per_mill
df_medical = df_cases_today.dropna(subset=disease_list)
colsList = ['location','total_deaths_per_million','life_expectancy','handwashing_facilities','hospital_beds_per
df_medical[colsList].sample(n=5)
```

Out[26]:

	location	total_deaths_per_million	life_expectancy	handwashing_facilities	hospital_beds_per_thousand	total_cases_per_mill
123	Namibia	59.427	63.71	44.600	NaN	5645.
138	Portugal	434.160	82.05	NaN	3.39	28911.
107	Moldova	567.680	71.90	86.979	5.80	26528.
105	Morocco	156.839	76.68	NaN	1.10	9585.
15	Bangladesh	40.130	72.59	34.808	0.80	2807.

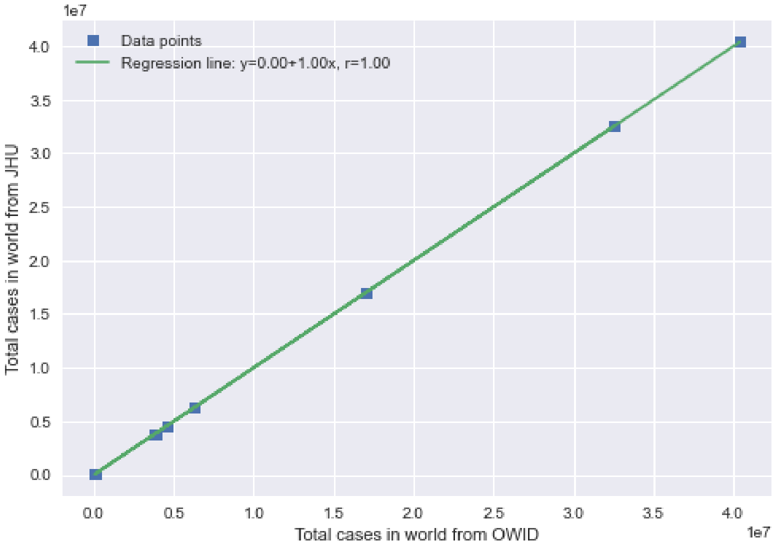
Assessing Data Consistency for JHU and OWID

Here we check for the consistancy of data retrived from two separate sources namely John Hopkins University and OWID. We perform the test for Total cases across the globe from both the data sets. We apply regression to see how closely the data at hand are related to eachother.

Taking total cases world wide using OWID data:

In [27]:
```
data = dailyKeyReturnerUsingLoc('World','total_cases')
x = []
for i in range(10):
    x.append(np.random.randint(313)-1)
data_JHU = []
data_OWID = []
for i in x:
    data_OWID.append(data[1][i])
    data_JHU.append(totalCasesInWorldJHU[i])

plt.style.use('seaborn')
slope, intercept, r, p, stderr = scipy.stats.linregress(data_OWID, data_JHU )
line = f'Regression line: y={intercept:.2f}+{slope:.2f}x, r={r:.2f}'
print(line)
fig, ax = plt.subplots()
ax.plot(data_OWID, data_JHU, linewidth=0, marker='s', label='Data points')
temp = []
for x in data_OWID:
    temp.append(slope*x)
ax.plot(data_OWID, intercept + temp, label=line)
ax.set_xlabel('Total cases in world from OWID')
ax.set_ylabel('Total cases in world from JHU')
ax.legend(facecolor='white')
plt.show()
```

```
Regression line: y=0.00+1.00x, r=1.00
```



Since the regression line has a slope of 1, we can well conclude that the given data sets are well consistant with eachother.

Quality Observations of the data:

- Validity: Some observations/rows in dataframes 'df_JHU_confirmed', 'df_JHU_fatality' contain the values for a region, for example Australia appears multiple times in column country as the observations are per region.
- Consistency: Some countries are referred to with varying names, for example 'US' and 'United States'. Other names are not valid.

Representation and Interpretation of Data:

We have plotted these graphs to show the ramifications of COVID-19 around the world and in India which is the most effected nation second to only the USA around the world. We have tried to highlight the nuances of the number COVID cases for each day since it got off the ground and have in the process exhibited a general trend between worldwide numbers and in India. The general plots are:

Scenario at Indian and Global levels:

1. Total cases and deaths worldwide due to COVID-19
2. Heat map of the total cases in the world
3. Total cases,deaths in India

Hypothesis using the data sets at hand:

Here we try to eshtablish some of the insconpicuous trends which do not feature in the mainstream news. We first hypothesise something which we feel could influence the impact of COVID-19 for that region/age group/race/people with a specific illness etc. With the help of our code and the gathered data we make plots and eshtablish a trend if the trend is present. If not, then we negate the hypothesis. We take into consideration some degree of error and have a confidence level above which we can safely say that the real time data supports our hypothesis. Our hypothesis are:

1. Comaprision of scenarios between India and few other countries
2. Effect of Age groups on number of covid cases
3. Effect of other diseases upon the cases and fatalities of Covid across several countries
4. Effect of altitude upon the fatalities of Covid across several countries
5. Effect of a country's economy upon the cases and fatalities due to covid
6. Effect of Hospital facilities upon the cases observed in covid
7. Effect of Covid cases on the Indian Economy
8. India vs USA GDP comaprision

Interpretation using bivariate data:

Here we in the last leg of our project by drawing out the final plots of those hypothesis which are supported by the data we have obtained. We deduce the proportoinality and the relation and conclude the project. The hypothesis whic have definite trends are:

9. Female smokers, age 70 older, deaths := scatter and heat map

Utility functions for plotting used :

In [28]:
```
def dayWisePlotter(x,y,figsize,xlabel,ylabel,title,MaxNLocator,xticklabels,rotation):
    plt.figure(figsize=figsize)
    ax = plt.axes(xlabel=xlabel,ylabel=ylabel,title=title)
    ax.xaxis.set_major_locator(plt.MaxNLocator(MaxNLocator))
    fig = ax.figure
    ax.figure.canvas.draw()
    ax.yaxis.set_major_formatter('{x:0.0f}')
    ax.set_xticklabels(xticklabels)
    fig.autofmt_xdate(rotation=rotation)
    ax.plot(x,y)
    plt.show()

def monthWisePlotter(x,y,figsize,xlabel,ylabel,title):
    plt.figure(figsize=figsize)
    ax = plt.axes(xlabel=xlabel,ylabel=ylabel,title=title)
    ax.plot(x,y)
    plt.show()
```

utility functions for interpretation:

In [29]:
```
def correlationChecker(data_OWID,data_JHU,owid,jhu):
    slope, intercept, r, p, stderr = scipy.stats.linregress(data_OWID, data_JHU )
    line = f'Regression line: y={intercept:.2f}+{slope:.2f}x, r={r:.2f}'
    print(line)
    fig, ax = plt.subplots()
    ax.plot(data_OWID, data_JHU, linewidth=0, marker='s', label='Data points')
    temp = []
    for x in data_OWID:
        temp.append(slope*x)
    ax.plot(data_OWID, intercept + temp, label=line)
    ax.set_xlabel(owid)
    ax.set_ylabel(jhu)
    ax.legend(facecolor='white')
    plt.show()
```

In [30]:
```
def bin(dataframe, cols):
# Create new columns that store the binned data
    for col in cols:
        new_col_name = "{}_bins".format(col)
        dataframe[new_col_name] = pd.qcut(dataframe[col], 10, labels=["1=10%", "2=20%", "3=30%", "4=40%", "5=50
```

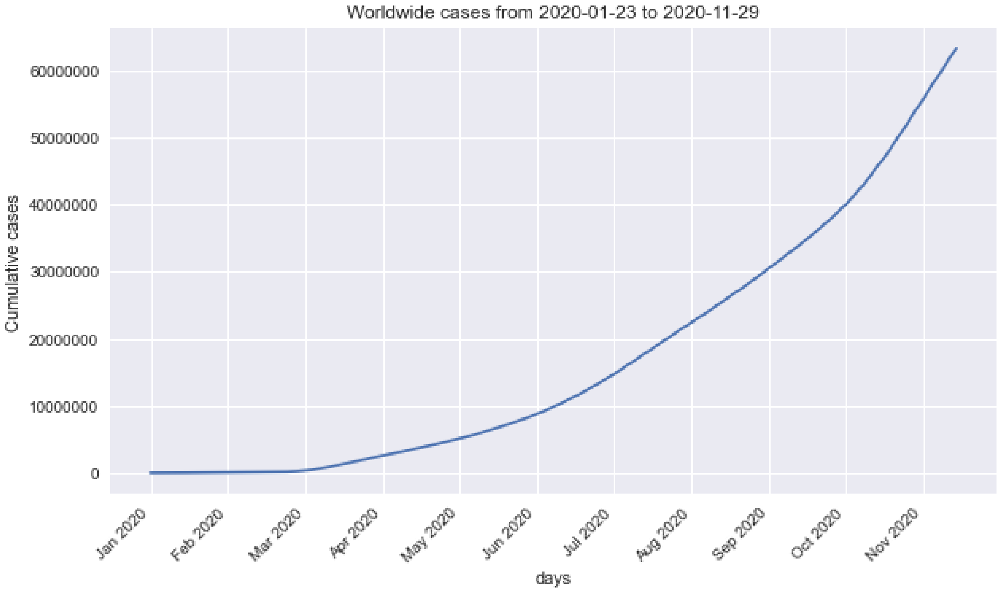
Current Scenario at Global and Indian level

Total cases and deaths worldwide due to COVID-19

Total cases world wide

In [31]:
```
xticklabels=["","Jan 2020","Feb 2020","Mar 2020","Apr 2020","May 2020","Jun 2020","Jul 2020","Aug 2020","Sep 20
dayWisePlotter(dateTotalCasesInWorldJHU,totalCasesInWorldJHU,[10,6],"days","Cumulative cases","Worldwide cases
```

```
<ipython-input-28-8077a346ca64>:8: UserWarning: FixedFormatter should only be used together with FixedLocato
r
  ax.set_xticklabels(xticklabels)
```

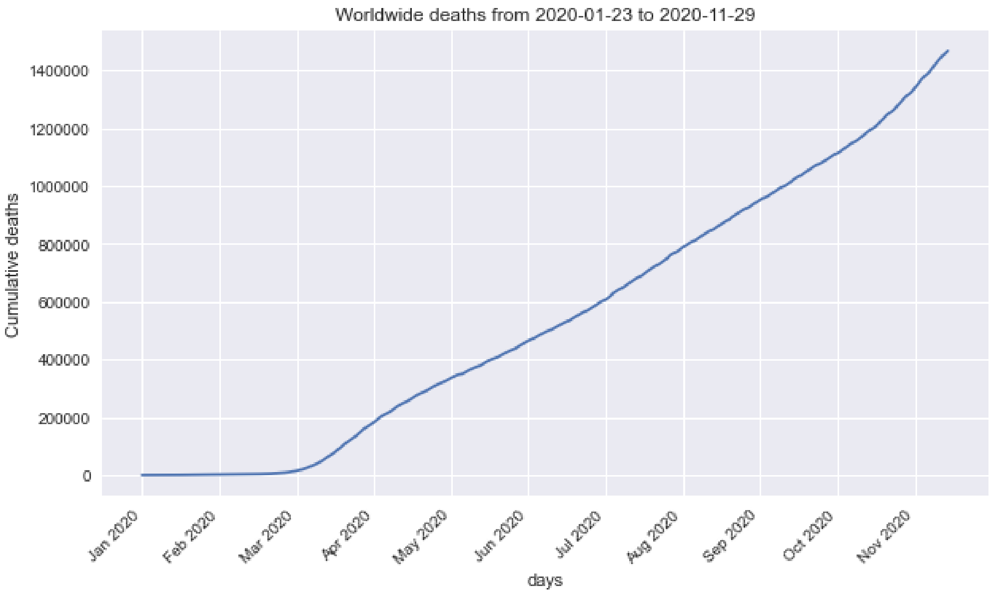


The first plot shows the number of cases worldwide on the y-axis verses the date on the x-axis with date starting from 23rd Jan,2020 to 29th Nov,2020. We can see that the number of cases is rising very drastically after end of March and is continuing its uptrend. The graph is not exponential but resemble to be exponential. In the python code we have taken an array named xticklables which contains the month and year, then we have used a function dayWisePlotter to plot the line graph for the same.

Total Deaths World Wide

In [32]:
```
dayWisePlotter(dateTotalDeathsInWorldJHU,totalDeathsInWorldJHU,[10,6],"days","Cumulative deaths","Worldwide dea
```

```
<ipython-input-28-8077a346ca64>:8: UserWarning: FixedFormatter should only be used together with FixedLocato
r
  ax.set_xticklabels(xticklabels)
```

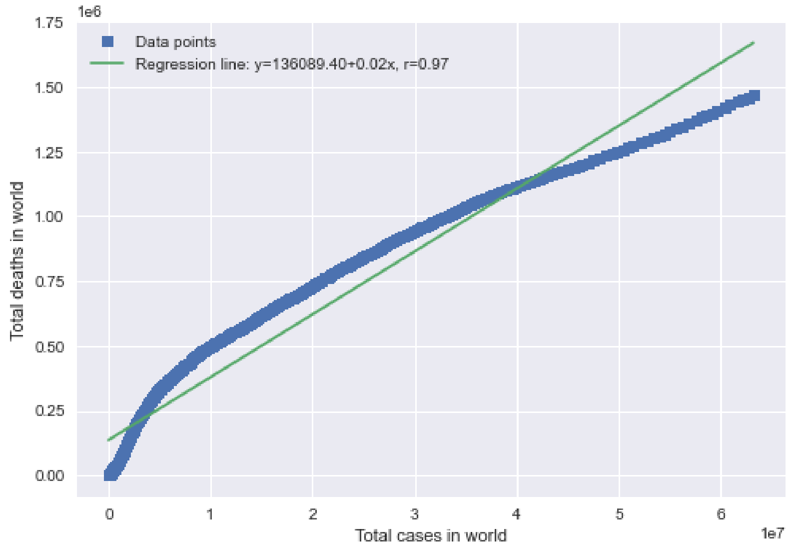


The second plot shows the number of deaths worlwide on the y-axis verses the date on the x-axis. Date starting from 23rd Jan,2020 to 29th Nov,2020. We can again observe that the number of deaths is rising very drastically after the end of March and is continuing its uptrend. The graph is not linear but resembles to be linear. In the python code we just called the function dayWisePlotter and changed the arguments to the totalDeathsInWorldJHU.

Let us look for the correlation between the cases and deaths.

In [33]:
```
correlationChecker(totalCasesInWorldJHU,totalDeathsInWorldJHU,"Total cases in world","Total deaths in world")
```

```
Regression line: y=136089.40+0.02x, r=0.97
```

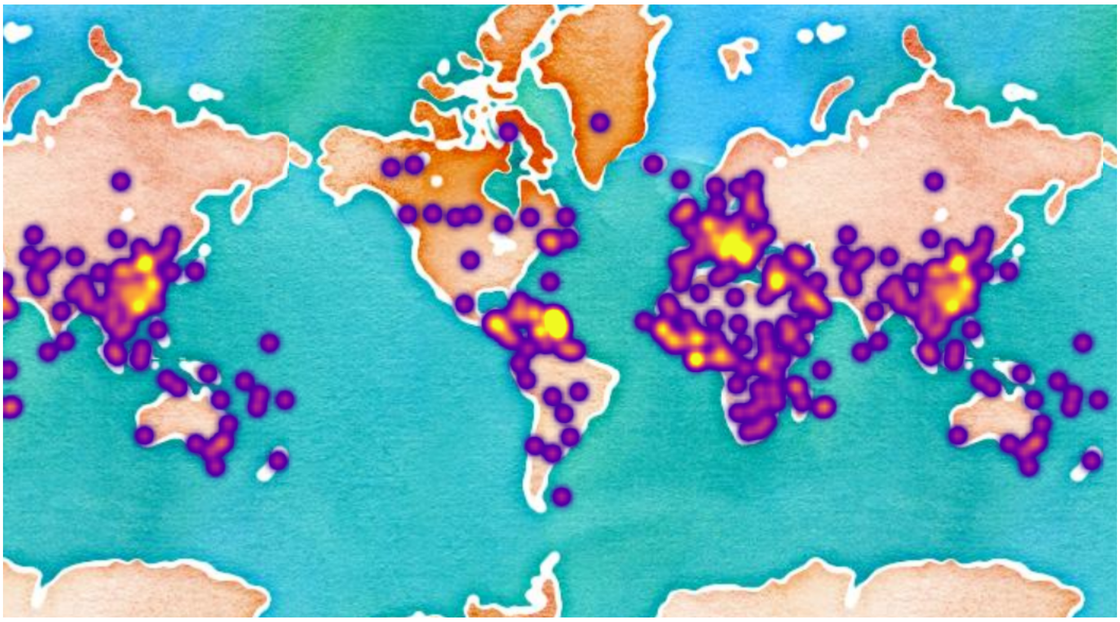


Now we have compared the number of cases and the number of deaths in India with the world. For this we are linear regression. In statistics, linear regression is a linear approach to modelling the relationship between a scalar response and one or more explanatory variables (also known as dependent and independent variables). The case of one explanatory variable is called simple linear regression; for more than one, the process is called multiple linear regression. In the first plot we have Total cases of covid in the world on y-axis verses the Total cases of covid in India on x-axis. In the second plot we have Total deaths due to covid in the world on y-axis verses the Total deaths due to covid in India on x-axis.

From the above **correlation coefficient = 0.97**
This means that there is a strong correlation and between total cases and total deaths.
From above graph we can say that country which reported higher number of cases will also report high number of deaths due to covid

Heat map of the total cases in the world

In [34]:
```
circle_radius = 10
typeLists = ['stamen-watercolor']
for i in typeLists:
    fig = px.density_mapbox(df_country_cases, lat='Lat', lon='Long', radius=circle_radius, zoom=0, mapbox_style
    fig.show()
```



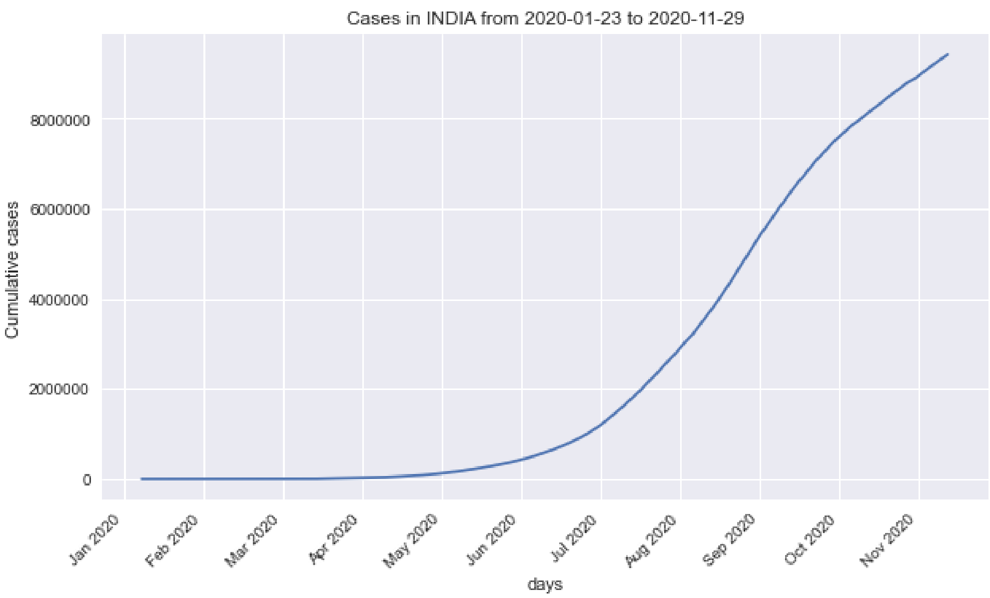
A heat map (or heatmap) is a data visualization technique that shows magnitude of a phenomenon as color in two dimensions. The variation in color may be by hue or intensity, giving obvious visual cues to the reader about how the phenomenon is clustered or varies over space. Here we have plotted a heat map for the total number of cases in the world. The program is written for the same.

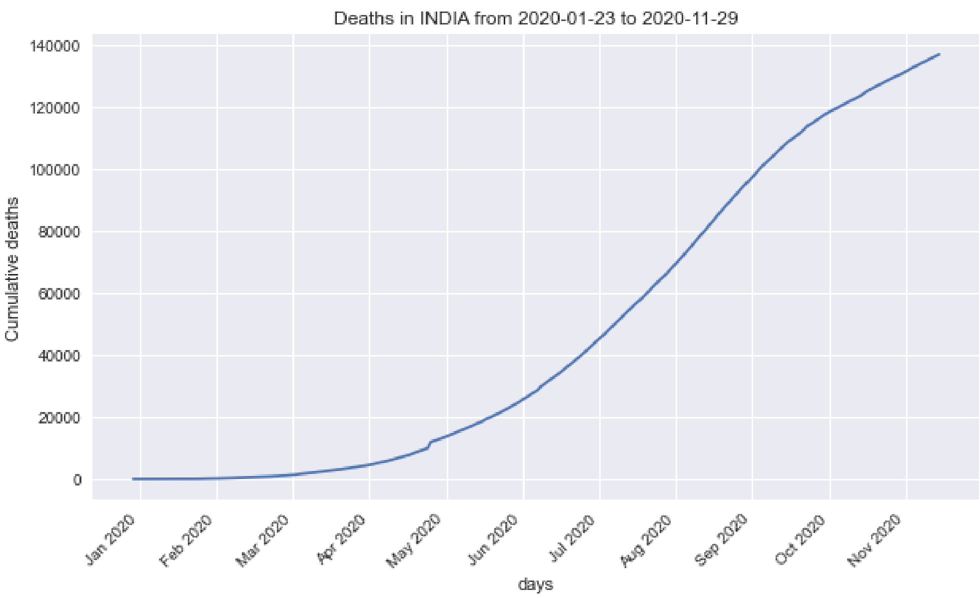
Total cases and Deaths in India

In [35]:

```
dataCasesIND = dailyKeyReturnerUsingLoc('India','total_cases')
dataCasesIND_month = []
for i in range(len(dataCasesIND[0])):
    test = dataCasesIND[0][i]
    test = test[len(test)-2:]
#     print(test)
    p = re.compile('01$')
    if p.match(test):
        dataCasesIND_month.append(dataCasesIND[1][i])
# print(dataCasesIND_month)
dataDeathsIND = dailyKeyReturnerUsingLoc('India','total_deaths')
dayWisePlotter(dataCasesIND[0],dataCasesIND[1],[10,6],"days","Cumulative cases","Cases in INDIA from 2020-01-23
dayWisePlotter(dataDeathsIND[0],dataDeathsIND[1],[10,6],"days","Cumulative deaths","Deaths in INDIA from 2020-0
```

```
<ipython-input-28-8077a346ca64>:8: UserWarning:

FixedFormatter should only be used together with FixedLocator
```





In this section we have plotted the graph for the number of cases and deaths in India. In this python program we have :

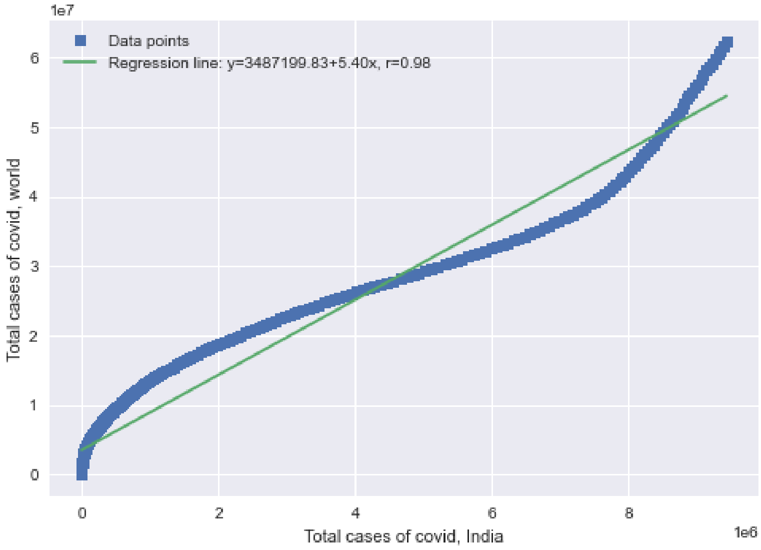
1. Taken input for the total number of cases in India in the dataCasesIND.
2. Taken data for the total number of deaths in India in the dataDeathsIND.

3. Then we have plotted the line graph for both using the function dayWisePlotter by providing suitable arguments.

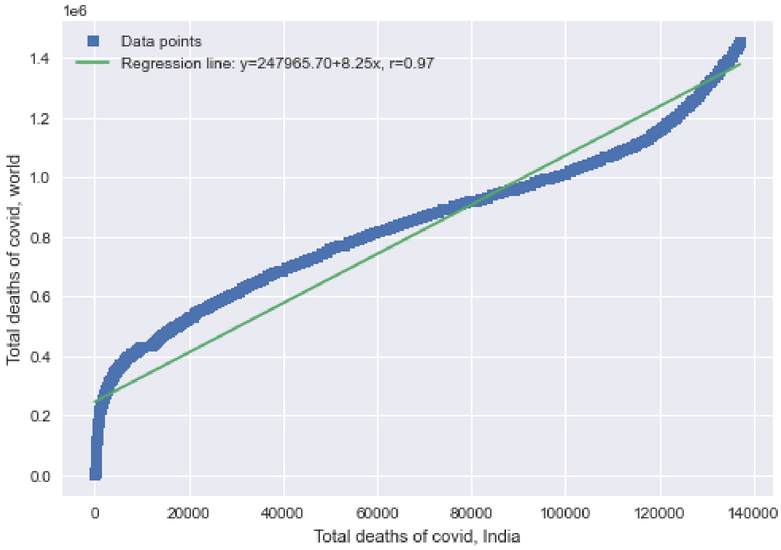
Let us check the correlation of cases and deaths in India as compared to those in the world

In [36]:
```
casesInIND = []
casesInWorld = []
for i in range(len(dataCasesIND[1])):
    if dataCasesIND[1][i]!= None and totalCasesInWorldJHU[i]!= None:
        casesInIND.append(dataCasesIND[1][i])
        casesInWorld.append(totalCasesInWorldJHU[i])
deathsInIND = []
deathsInWorld = []
for i in range(len(dataDeathsIND[1])):
    if dataDeathsIND[1][i]!= None and totalDeathsInWorldJHU[i]!= None:
        deathsInIND.append(dataDeathsIND[1][i])
        deathsInWorld.append(totalDeathsInWorldJHU[i])



correlationChecker(casesInIND,casesInWorld,"Total cases of covid, India","Total cases of covid, world")
correlationChecker(deathsInIND,deathsInWorld,"Total deaths of covid, India","Total deaths of covid, world")
```

```
Regression line: y=3487199.83+5.40x, r=0.98
```




```
Regression line: y=247965.70+8.25x, r=0.97
```



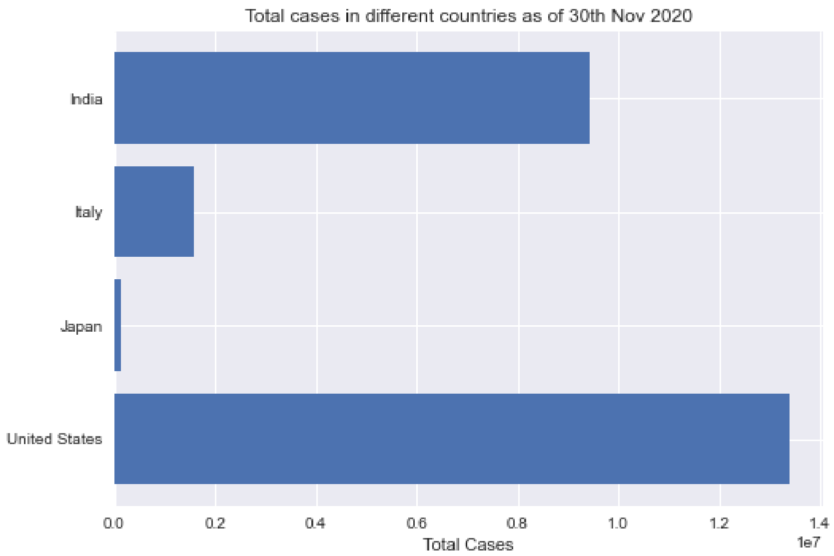
For both the plots correlation coefficient is almost same 0.98
which tells us that as cases and deaths in India are increasing simultaneously does cases and deaths in world.But there is some difference in the rate
As cases increase by 1 unit in india world cases increase by 5.40 units and as deaths increase by 1 unit in India world deaths increase by 8.25
which indiacates that mortality

Total cases in selected countries

In [37]:

```
fig, ax = plt.subplots()
y_pos = np.arange(len(df_cases_selected_countries["location"]))
ax.barh(y_pos, df_cases_selected_countries["total_cases"])
ax.set_yticks(y_pos)
ax.set_yticklabels(df_cases_selected_countries["location"])
ax.invert_yaxis()  # labels read top-to-bottom
ax.set_xlabel('Total Cases')
ax.set_title('Total cases in different countries as of 30th Nov 2020')
```

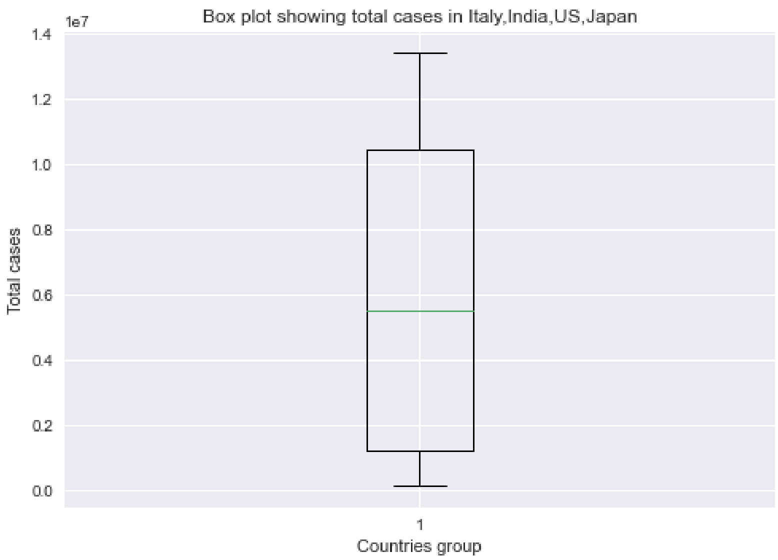
Out[37]: Text(0.5, 1.0, 'Total cases in different countries as of 30th Nov 2020')



Box plot representation of the above data

In [38]:

```
plt.xlabel("Countries group")
plt.ylabel("Total cases")
plt.title("Box plot showing total cases in Italy,India,US,Japan")
plt.boxplot(df_cases_selected_countries["total_cases"])
plt.show()
```

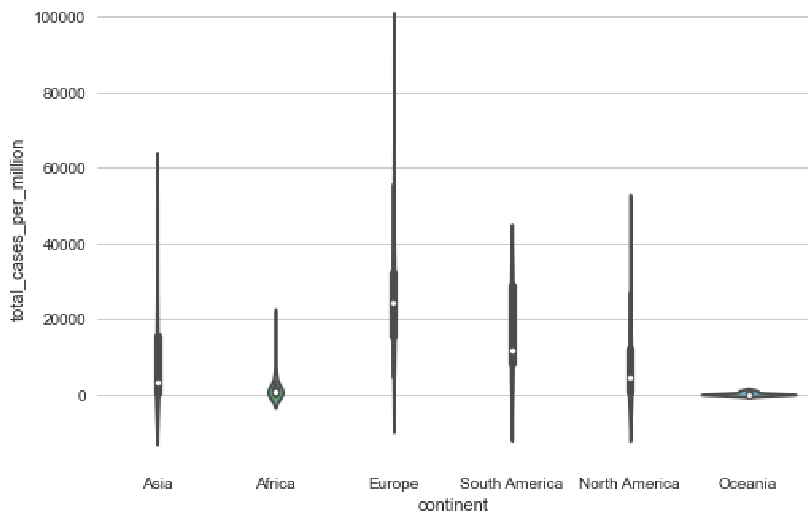


Total cases and deaths in world by continents

In [39]:

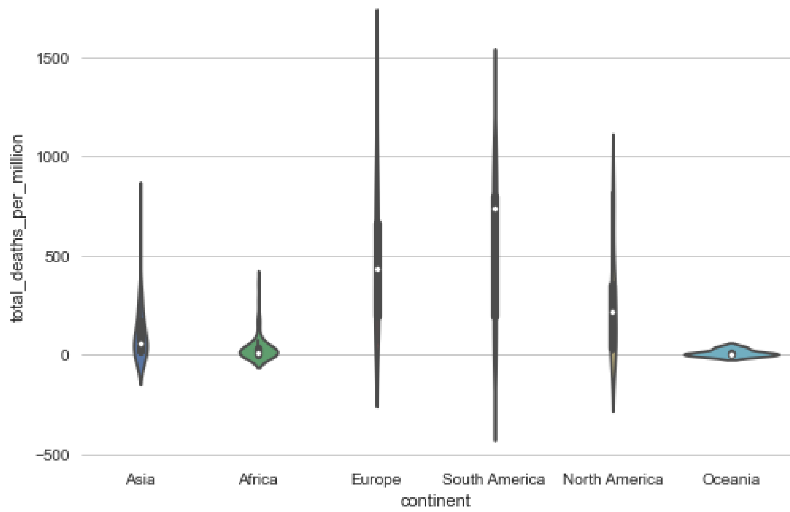
```
print("Total cases")
sns.set_style('whitegrid')
sns.violinplot(x='continent', y='total_cases_per_million', data=df_cases_today)
plt.show()
print("Total deaths")
sns.violinplot(x='continent', y='total_deaths_per_million', data=df_cases_today)
```

```
Total cases
```




```
Total deaths
```

Out[39]:
```
<AxesSubplot:xlabel='continent', ylabel='total_deaths_per_million'>
```



Hypothesis

Cases Across Different Countries :

Hypothesis 1: Total cases in India is related to that of USA

H0:Total cases in India are same as total cases in USA

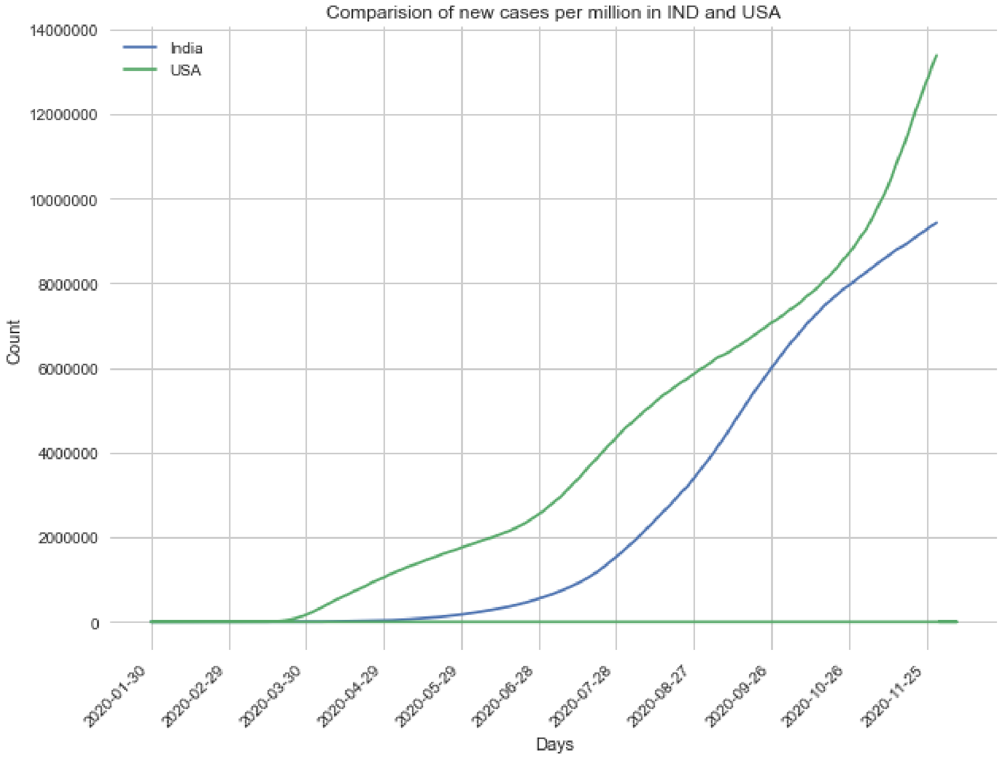
H1:Total cases in India are not same as total cases in USA

Analysing Total cases in India and USA

We shall fist begin with analysing the cases of COVID that were reported in 2020 in India and USA. For this we have plotted the line graph of new cases per million in both the countries in the same chart. We have used the attribute per million to account for the difference in population between the countries.

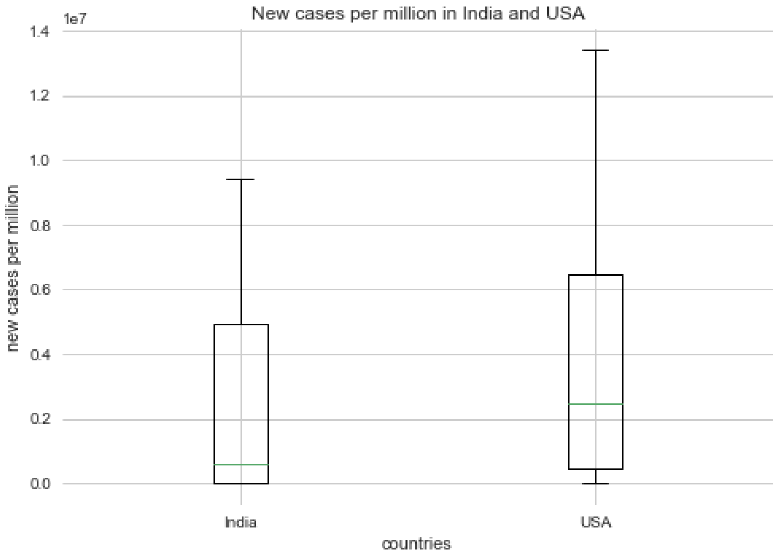
In [40]:
```
df_covid_ind = df_covid_data.loc[df_covid_data['location']=='India'].dropna(subset = ['total_cases'])
df_covid_usa = df_covid_data.loc[df_covid_data['location']=='United States'].dropna(subset = ['total_cases'])
```

In [41]:
```
plt.figure(figsize=[10,8])
ax = plt.axes(xlabel="Days",ylabel="Count",title="Comparision of new cases per million in IND and USA")
ax.xaxis.set_major_locator(plt.MaxNLocator(12))
fig = ax.figure
ax.figure.canvas.draw()
ax.yaxis.set_major_formatter('{x:0.0f}')
fig.autofmt_xdate(rotation=45)
ax.plot(df_covid_ind['date'],df_covid_ind['total_cases'],label="India")
ax.plot(df_covid_usa['date'],df_covid_usa['total_cases'],label="USA")
plt.legend()
plt.show()
```



Box Plot

In [42]:
```
data=[df_covid_ind['total_cases'],df_covid_usa['total_cases']]
plt.boxplot(data,notch=0, sym='+')
plt.xticks([1, 2], ['India', 'USA'])
plt.title("New cases per million in India and USA")
plt.xlabel("countries")
plt.ylabel("new cases per million")
plt.show()
```



In [43]:
```
print(pearsonr(df_covid_ind['total_cases'],df_covid_usa['total_cases'][8:]))
```

```
(0.9639335741043054, 3.0657370103616465e-176)
```

The P value of the statistic is found to be very small and equal to $3x10^{-176}$, which is less than 0.05. Hence we reject H0.

Conclusion : From the above plots it is clear that total cases in India is strongly correlated (r=0.9639335741043054)to the total cases in USA.They are not same but there is strong correlation because of some underlying factor.

Hypothesis 2: Total cases and deaths in a country directly depends on the population of the country

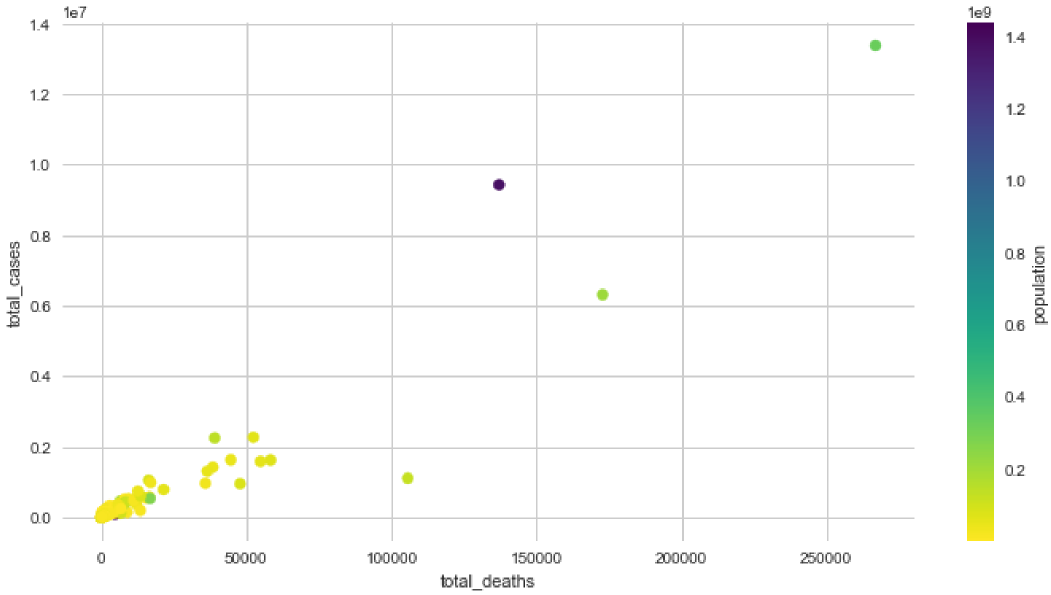
H0:Cases and deaths are independent of population(not related)

H1:Covid cases and deaths depend on population(related)

Scatter plot of cases, deaths due to COVID along with heat map of population

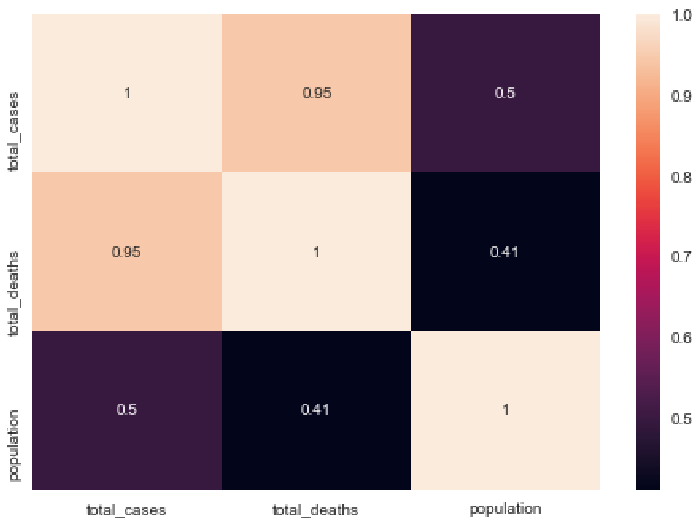
In [44]:
```
df_cases_today_300 = df_cases_today.loc[df_cases_today['total_cases'] > 0][:-2]
```

```
plt.figure(figsize=[12,6])
plt.scatter(data=df_cases_today_300, x='total_deaths', y='total_cases', c='population', cmap='viridis_r' );
plt.colorbar(label='population');
plt.xlabel('total_deaths');
plt.ylabel('total_cases');
axes = plt.gca()
```



The above scatter plot does support H1, however for a clearer picture, we shall plot the heat map of the correlation matrix of the attributes.

In [45]:
```
df_cases_today_300 = df_cases_today_300.dropna(subset = ['total_cases','total_deaths','population'])
correlation_mat = df_cases_today_300[['total_cases','total_deaths','population']].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
print("Assoc. - population and total_cases",pearsonr(df_cases_today_300['population'],df_cases_today_300['total
print("Assoc. - population and total_deaths",pearsonr(df_cases_today_300['population'],df_cases_today_300['tota
```




```
Assoc. - population and total_cases (0.49735615304133607, 2.923728984845876e-12)
Assoc. - population and total_deaths (0.4107516845853533, 1.8064893291618904e-08)
```

The positive correlation and extremely small P-values values do suggest that total cases and deaths reported of COVID in a country do depend on population directly and hence H0 is rejected.

Effect of Age groups on number of covid cases

In this section we tried to find the relation of the number of cases with the age group. We have shown this relation through the help of Pie Chart. A pie chart (or a circle chart) is a circular statistical graphic, which is divided into slices to illustrate numerical proportion. In a pie chart, the arc length of each slice (and consequently its central angle and area), is proportional to the quantity it represents.

In the python code:

i)Through the help of the loop we have appended the age in the array ageGroup[].

ii)Through the help of the loop we have appended the percentage in the array percentage[].

iii)Then we have used the function plt.pie to plot in pie chart using the respective arguments.

Hypothesis 1: Proportion of COVID cases in India is directly proportional to age upto 75 years of age

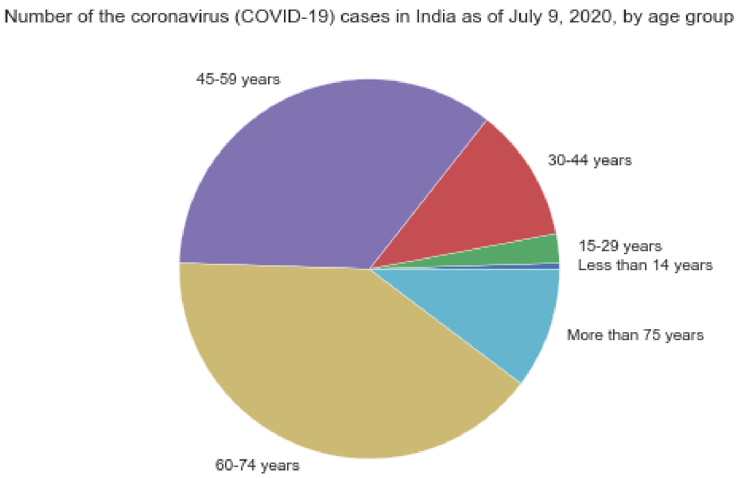
H0: Number of cases are same across all age groups in India

H1: There is some difference in number of cases across the age groups

Pie chart depicting the proportions of cases in various age groups in India

In [46]:
```
ageGroup = []
percentage = []
colAge = ws_age_data['B']
colPer = ws_age_data['C']
for x in range(5,11):
    ageGroup.append(colAge[x].value)
    percentage.append(colPer[x].value)
plt.title("Number of the coronavirus (COVID-19) cases in India as of July 9, 2020, by age group")
plt.pie(percentage, labels = ageGroup)
```

Out[46]:
```
([<matplotlib.patches.Wedge at 0x2e8c64feb80>,
  <matplotlib.patches.Wedge at 0x2e8c64fefd0>,
  <matplotlib.patches.Wedge at 0x2e8c650c460>,
  <matplotlib.patches.Wedge at 0x2e8c650c880>,
  <matplotlib.patches.Wedge at 0x2e8c650cc40>,
  <matplotlib.patches.Wedge at 0x2e8c651d0a0>],
 [Text(1.099864295735893, 0.017278048656840017, 'Less than 14 years'),
  Text(1.0933570509441903, 0.12070774271198717, '15-29 years'),
  Text(0.9397051872092018, 0.5717990566030334, '30-44 years'),
  Text(-0.4652279531565006, 0.9967762796143441, '45-59 years'),
  Text(-0.36610163557183434, -1.0372895412716876, '60-74 years'),
  Text(1.042912059127538, -0.34976340135348516, 'More than 75 years')])
```



In [47]:
```
df_age_data_modif = df_age_data.loc[5:10,1:2]
df_age_data_modif['age'] = [5,14,25,40,70,90]
df_age_data_modif['per'] = df_age_data_modif[2]
df_age_data_modif
```

Out[47]:

	1	2	age	per
5	Less than 14 years	0.5	5	0.5
6	15-29 years	2.5	14	2.5
7	30-44 years	11.4	25	11.4
8	45-59 years	35.1	40	35.1
9	60-74 years	40.2	70	40.2
10	More than 75 years	10.3	90	10.3

In [48]:
```
print("Assoc. - age and proportion of cases",pearsonr(df_age_data_modif['age'][:-1],df_age_data_modif['per'][:-
```

```
Assoc. - age and proportion of cases (0.9396159643637035, 0.017649963464187568)
```

p value = 0.017649963464187568 < 0.05 buut greater than 0.01 So we reject H0 at 5% level but Fail to reject at 1% level. For our project we have kept the significance level as 5% and hence H0 is rejected.

Hypothesis 2: Countries with a higher proportion of population older than 65 years of age, had greater cases and deaths per million

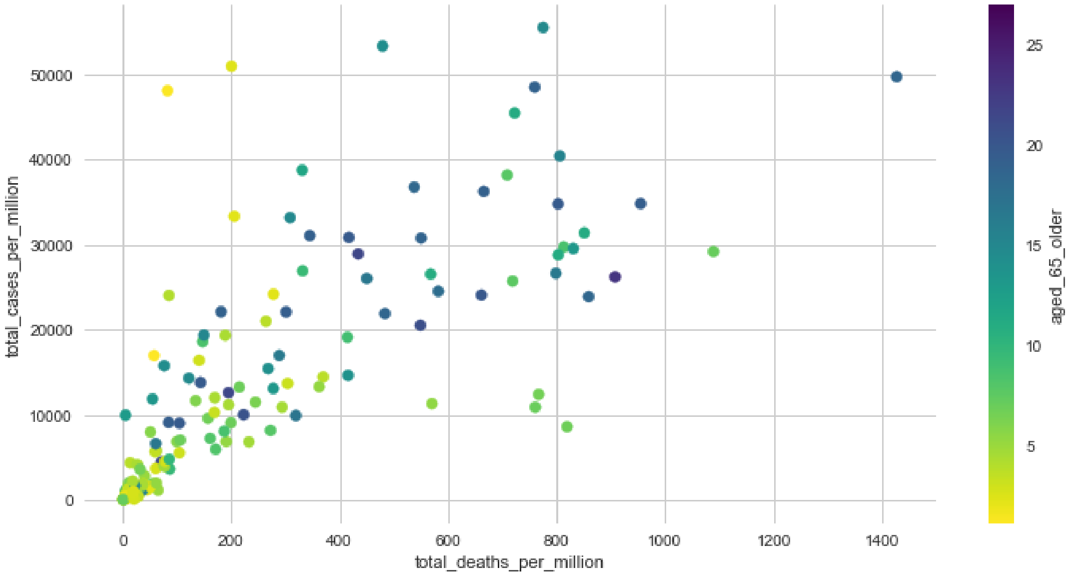
H0:Cases in a country do not depend on proportion of old people aged >65

H1:Cases in a country with more proportion of old people are greater than the country with less proportion

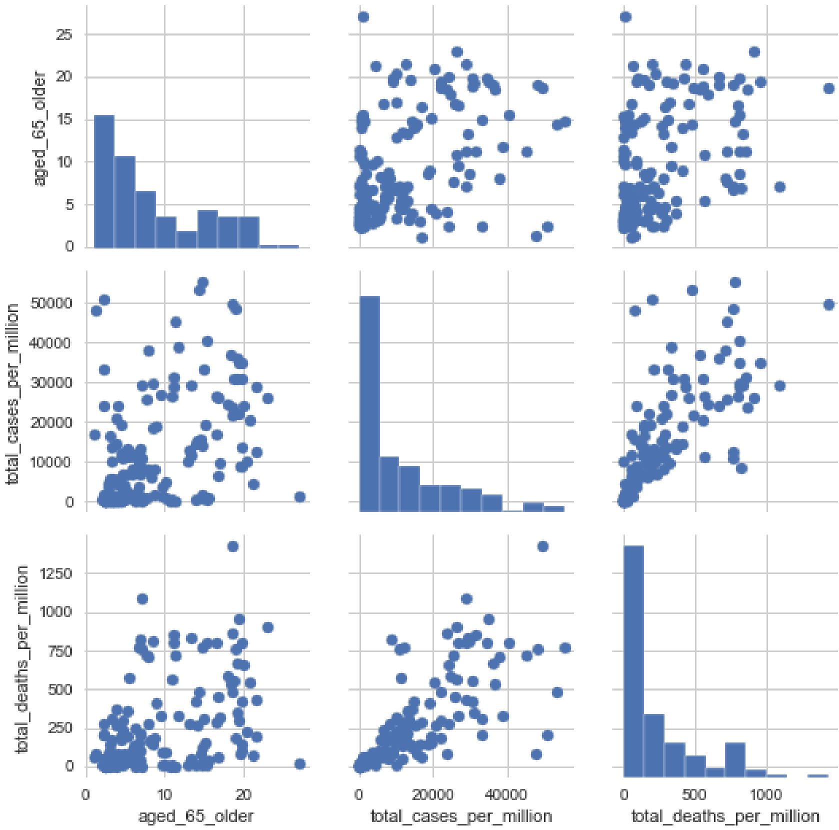
In [49]:
```
ageGroupList = ['total_cases_per_million', 'aged_70_older', 'aged_65_older', 'median_age', 'total_deaths_per_mi
```

Proportion of population older than 65 years of age

In [50]:
```
plt.figure(figsize=[12,6])
plt.scatter(data=df_ageGroupData, x='total_deaths_per_million', y='total_cases_per_million', c='aged_65_older',
plt.colorbar(label='aged_65_older');
plt.xlabel('total_deaths_per_million');
plt.ylabel('total_cases_per_million');
axes = plt.gca()
```

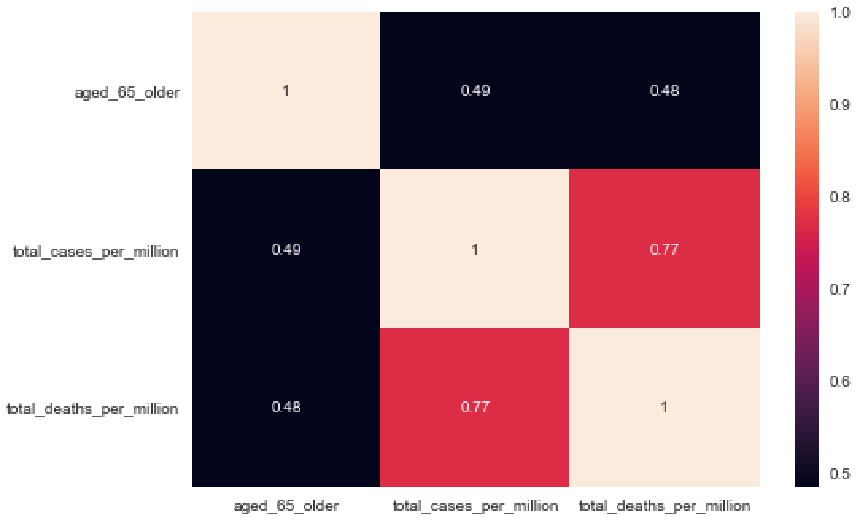


In [51]:
```
g=sb.PairGrid(data=df_ageGroupData, vars=[ 'aged_65_older','total_cases_per_million','total_deaths_per_million'
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [52]:
```
correlation_mat = df_ageGroupData[[ 'aged_65_older','total_cases_per_million','total_deaths_per_million']].corr
```

```
sns.heatmap(correlation_mat, annot = True)

plt.show()
```



In [53]:
```
df_ageGroupData = df_ageGroupData.dropna(subset = ['aged_65_older','total_cases_per_million','total_deaths_per_
print("Assoc. - aged_65_older and total_cases_per_million",pearsonr(df_ageGroupData['aged_65_older'],df_ageGrou
print("Assoc. - aged_65_older and total_deaths_per_million",pearsonr(df_ageGroupData['aged_65_older'],df_ageGro
```

```
Assoc. - aged_65_older and total_cases_per_million (0.4727457357576012, 1.2667519752367446e-10)
Assoc. - aged_65_older and total_deaths_per_million (0.4842577583676207, 3.8358818068327264e-11)
```

The p value obtained for both cases and deaths 1.2667519752367446e-10,3.8358818068327264e-11 are highly lesser than 0.05

So we reject H0

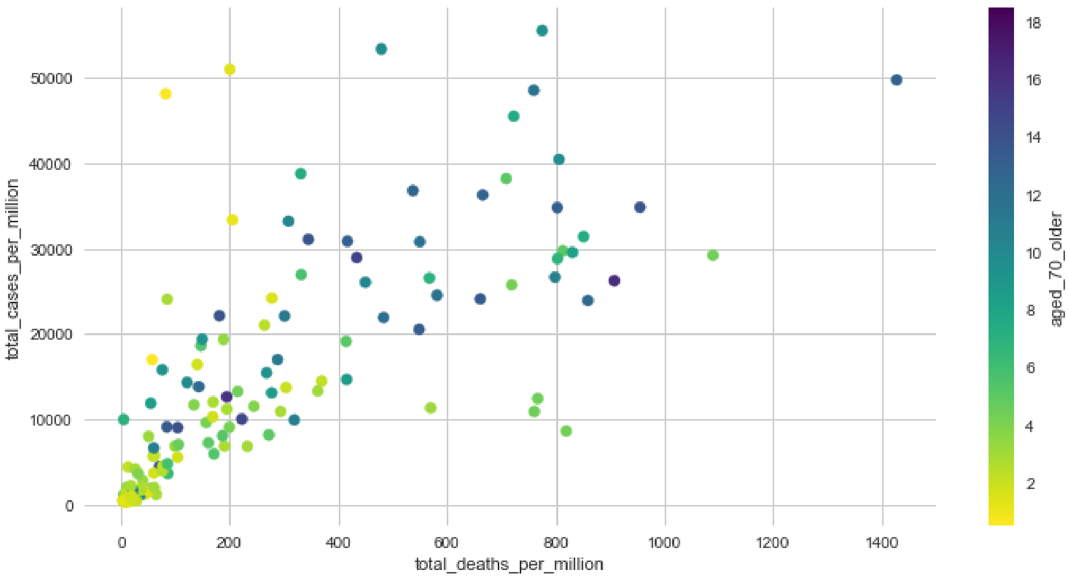
Hypothesis 3: Countries with a higher proportion of population older than 70 years of age had greater cases and deaths per million

H0:Cases and deaths in a country are independent of proportion of population older than 70 years

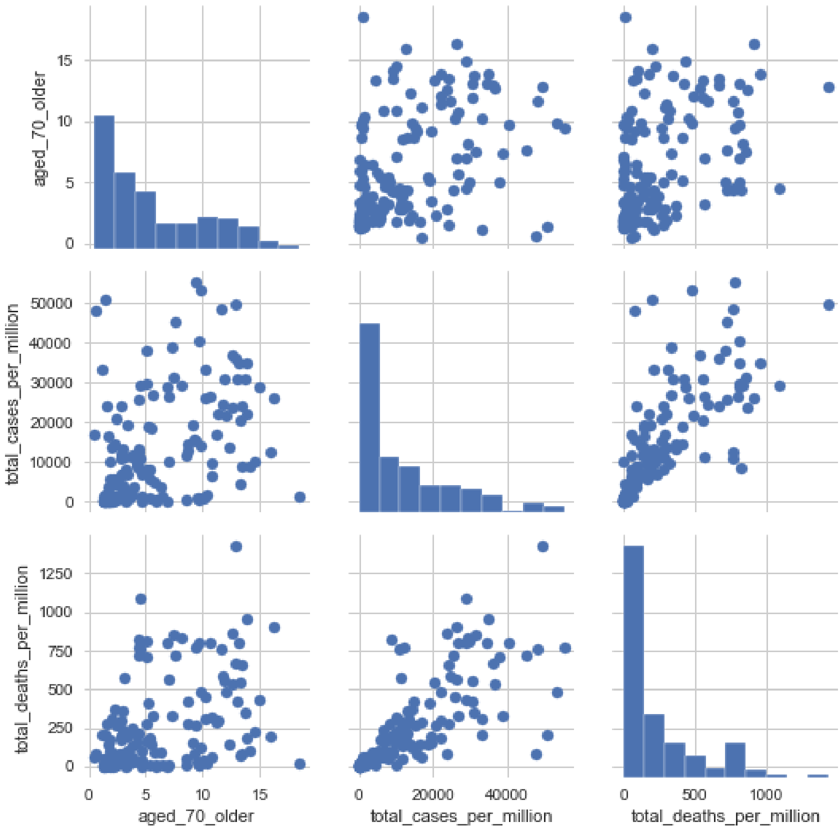
H1:Cases and deaths in a country depend on proportion of population older than 70 years

Proportion of population older than 70 years of age

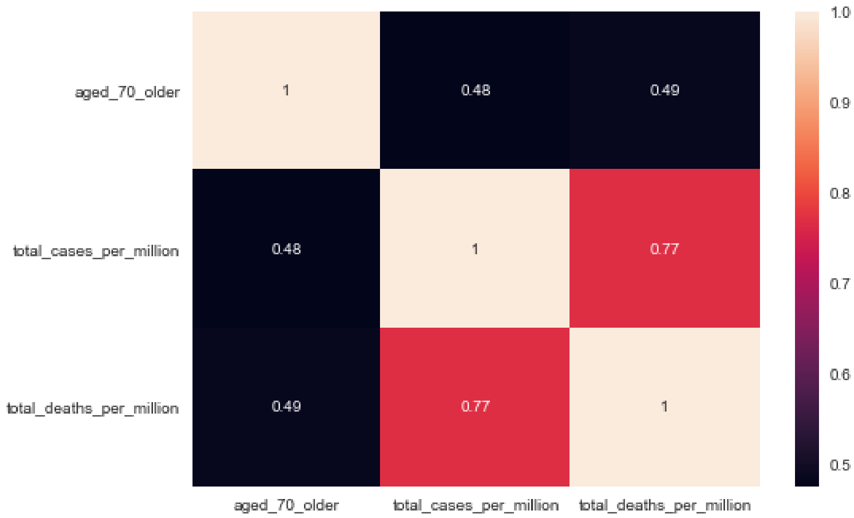
In [54]:
```
plt.figure(figsize=[12,6])
plt.scatter(data=df_ageGroupData.loc[df_ageGroupData['total_cases_per_million'] > 300], x='total_deaths_per_mil
plt.colorbar(label='aged_70_older');
plt.xlabel('total_deaths_per_million');
plt.ylabel('total_cases_per_million');
axes = plt.gca()
```



In [55]:
```
g=sb.PairGrid(data=df_ageGroupData, vars=[ 'aged_70_older','total_cases_per_million','total_deaths_per_million'
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [56]:
```
correlation_mat = df_ageGroupData[[ 'aged_70_older','total_cases_per_million','total_deaths_per_million']].corr
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [57]:
```
df_ageGroupData = df_ageGroupData.dropna(subset = ['aged_70_older','total_cases_per_million','total_deaths_per_
print("Assoc. - aged_70_older and total_cases_per_million",pearsonr(df_ageGroupData['aged_70_older'],df_ageGrou
print("Assoc. - aged_70_older and total_deaths_per_million",pearsonr(df_ageGroupData['aged_70_older'],df_ageGro
```

```
Assoc. - aged_70_older and total_cases_per_million (0.4757728436054714, 9.292433091550181e-11)
Assoc. - aged_70_older and total_deaths_per_million (0.485546173743143, 3.3464817568300115e-11)
```

p value obtained are very low so we reject H0

Hypothesis 4: Countries with a higher higher median age had greater cases and deaths per million

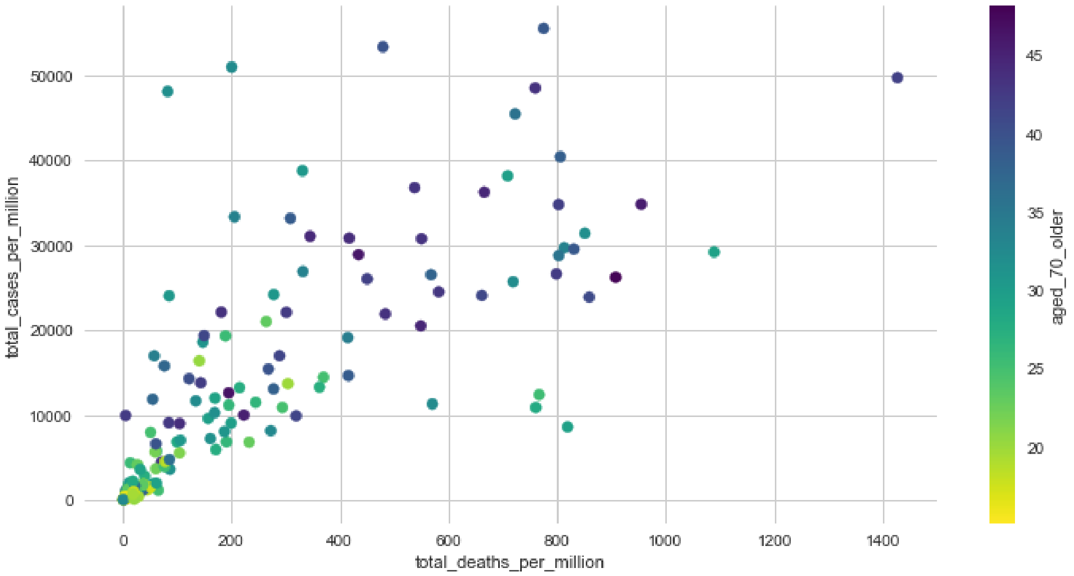
H0:There is no relation between median age of a country and cases and deaths per million in the country

H1:Countries with a higher higher median age had greater cases and deaths per million

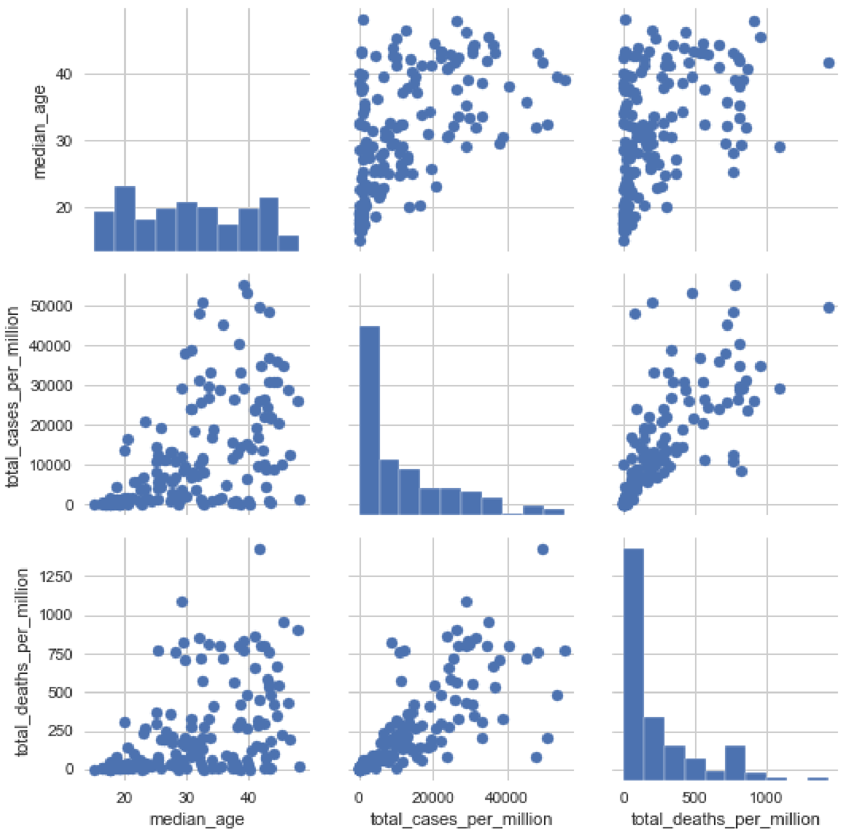
Median age

In [58]:
```
plt.figure(figsize=[12,6])
plt.scatter(data=df_ageGroupData, x='total_deaths_per_million', y='total_cases_per_million', c='median_age', cm
plt.colorbar(label='aged_70_older');
plt.xlabel('total_deaths_per_million');
```

```
plt.ylabel('total_cases_per_million');
axes = plt.gca()
```



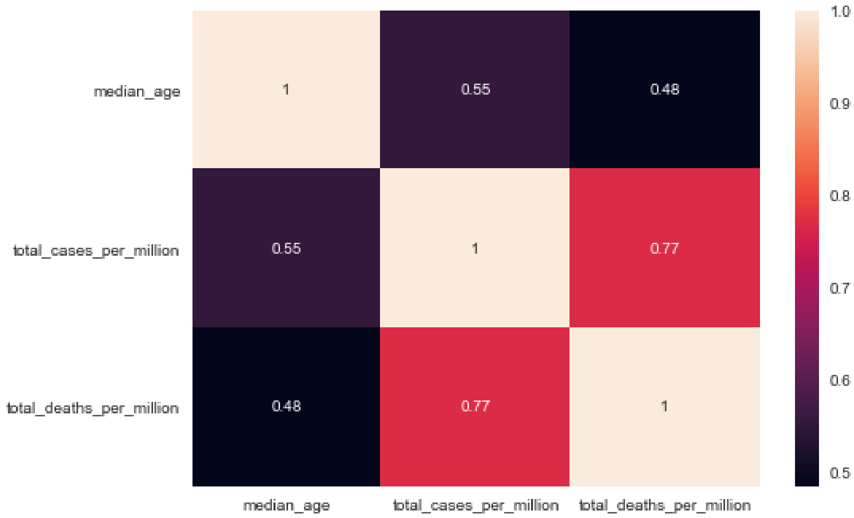
In [59]:
```
g=sb.PairGrid(data=df_ageGroupData, vars=[ 'median_age','total_cases_per_million','total_deaths_per_million'])
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



Next we have made the correlation matrix for the factors mentioned above. In this matrix if the values are above 0 then they are directly proportional. And if the values are less than 0 then they are inversely proportional. In the matrix we can see that all the diagonal elements are having value 1 as they are relation to each other only If the value is equal to 0 then it indicates that there is no relation between the two elements being compared.

Correlation matrix as a heat map of the above plots

In [60]:
```
correlation_mat = df_ageGroupData[[ 'median_age','total_cases_per_million','total_deaths_per_million']].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [61]:
```
df_ageGroupData = df_ageGroupData.dropna(subset = ['median_age','total_cases_per_million','total_deaths_per_mil
print("Assoc. - median_age and total_cases_per_million",pearsonr(df_ageGroupData['median_age'],df_ageGroupData[
print("Assoc. - median_age and total_deaths_per_million",pearsonr(df_ageGroupData['median_age'],df_ageGroupData
```

```
Assoc. - median_age and total_cases_per_million (0.5545722142180631, 9.037026555671896e-15)
Assoc. - median_age and total_deaths_per_million (0.4848613094684803, 3.5985545595939865e-11)
```

p value obtained are very low so we reject H0

As we hypothesized, a strong positive correlation exists in between the amount of fatal cases per population and median_age, aged_65_older, aged_70_older. There is a positive correlation for each of the age related statistics to the amount of deaths per population.

Effect of altitude upon the fatalities of Covid across several countries

Hypothesis 1: Countries with a higher altitude had greater cases per million

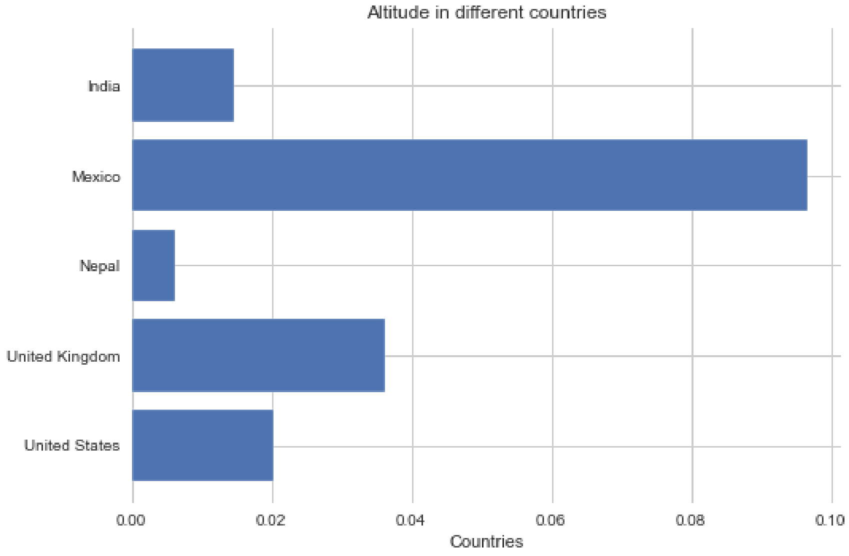
Hypothesis test:
H0: Cases are independent of altitude of the place
H1: There is some effect of altitude on cases at that place

In [62]:
```
countries = ['India',
             'Mexico',
             'Nepal',
             'United Kingdom',
             'United States']
```

In [63]:
```
df_altitude_modif_countries = df_altitude_modif.T
df_altitude_modif_countries = df_altitude_modif_countries[countries]
df_altitude_modif_countries = df_altitude_modif_countries.T
df_altitude_modif_countries['Country'] = countries
```

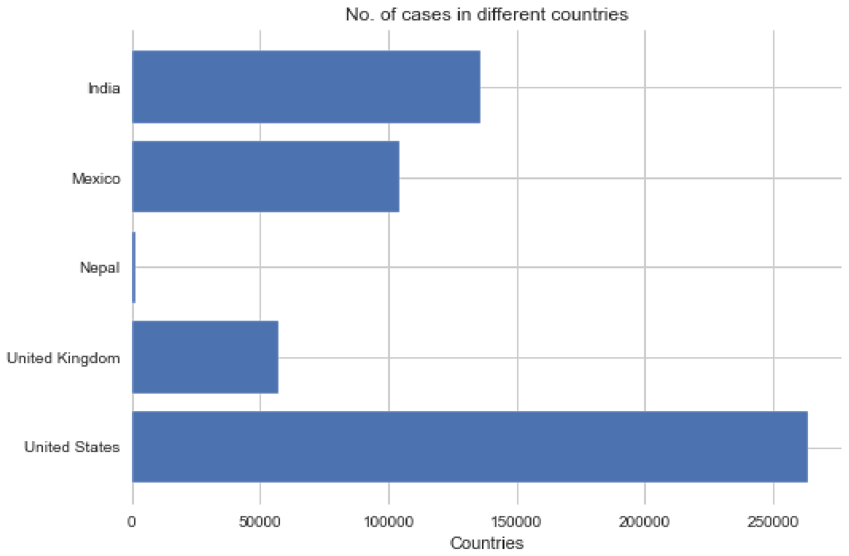
In [64]:
```
fig, ax = plt.subplots()
y_pos = np.arange(len(df_altitude_modif_countries["Country"]))
ax.barh(y_pos, df_altitude_modif_countries["Fatality ratio"])
ax.set_yticks(y_pos)
ax.set_yticklabels(df_altitude_modif_countries["Country"])
ax.invert_yaxis()  # labels read top-to-bottom
ax.set_xlabel('Countries')
ax.set_title('Altitude in different countries')
```

Out[64]:
```
Text(0.5, 1.0, 'Altitude in different countries')
```



In [65]:
```
fig, ax = plt.subplots()
y_pos = np.arange(len(df_altitude_modif_countries["Country"]))
ax.barh(y_pos, df_altitude_modif_countries["No of cases"])
ax.set_yticks(y_pos)
ax.set_yticklabels(df_altitude_modif_countries["Country"])
ax.invert_yaxis()  # labels read top-to-bottom
ax.set_xlabel('Countries')
ax.set_title('No. of cases in different countries')
```

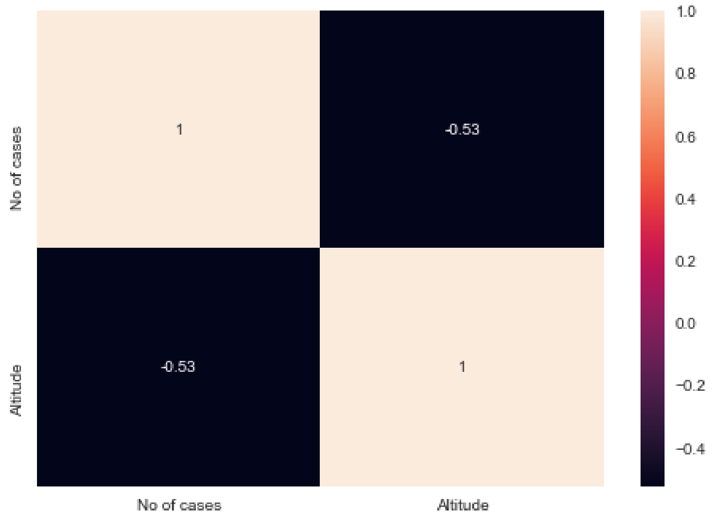
Out[65]: `Text(0.5, 1.0, 'No. of cases in different countries')`



Next we have made the correlation matrix for the factors mentioned above. In this matrix if the values are above 0 then they are directly proportional. And if the values are less than 0 then they are inversely proportional. In the matrix we can see that all the diagonal elements are having value 1 as they are relation to each other only If the value is equal to 0 then it indicates that there is no relation between the two elements being compared.

In [66]:
```
df_altitude_modif_countries['Altitude'] = df_altitude_modif_countries['Altitude'].map(lambda name : float(name)
df_altitude_modif_countries['No of cases'] = df_altitude_modif_countries['No of cases'].map(lambda name : float

correlation_mat = df_altitude_modif_countries[['No of cases','Altitude']].corr()
print(correlation_mat)
sns.heatmap(correlation_mat, annot = True)
plt.show()
```

```
             No of cases  Altitude
No of cases     1.000000 -0.525455
Altitude       -0.525455  1.000000
```



From the above we have that correlation between Altitude and number of cases is negative -0.53. The negative correlation depicts inverse relationship between the two factors. Let us take two countries India and Nepal. The altitude of nepal is higher compared to India which indicates number of cases is more in India than Nepal.

Effect of a country's economy upon the cases and fatalities due to covid

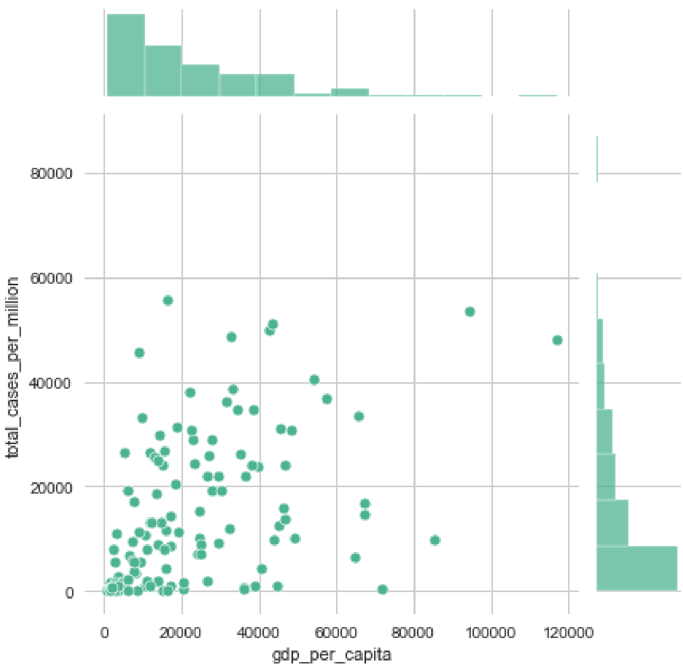
Hypothesis 1: Countries with higher GDP per capita reported more number of cases of COVID

H0:There is no dependency of cases on GDP per capita of a country

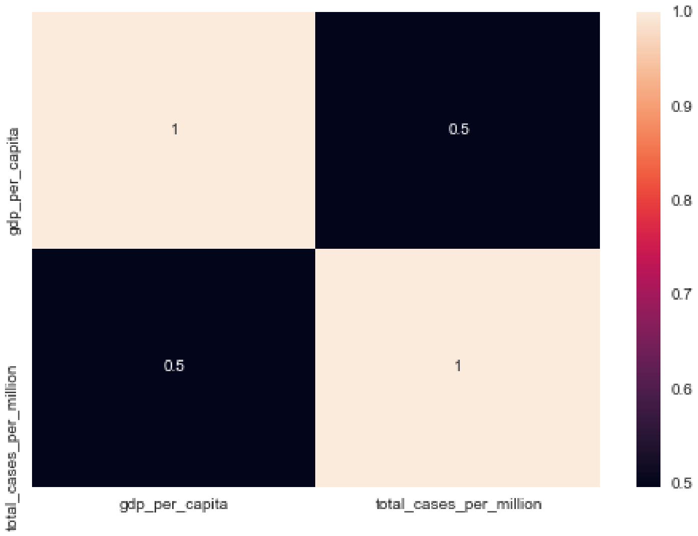
H1:Countries with higher GDP per capita reported more number of cases of COVID

In [67]:
```
sns.jointplot(x=df_economy['gdp_per_capita'], y=df_economy['total_cases_per_million'],  color="#4CB391")
```

Out[67]: <seaborn.axisgrid.JointGrid at 0x2e8c682b550>



In [68]:
```
correlation_mat = df_economy[['gdp_per_capita','total_cases_per_million']].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [69]:
```
df3 = df_economy.copy()
# This creates new columns filled with the binned column data
bin(df3, economy_list)
```

In [70]:
```
anova_df = df3[['gdp_per_capita_bins','total_cases_per_million']].dropna()
anova = smf.ols(formula='total_cases_per_million ~ C(gdp_per_capita_bins)', data=anova_df).fit()
print(anova.summary())
```

```
                                OLS Regression Results
===================================================================================
Dep. Variable:     total_cases_per_million   R-squared:                       0.324
Model:                                 OLS   Adj. R-squared:                  0.274
Method:                      Least Squares   F-statistic:                     6.458
Date:                     Fri, 11 Dec 2020   Prob (F-statistic):           1.82e-07
Time:                             12:50:16   Log-Likelihood:                -1413.9
No. Observations:                      131   AIC:                             2848.
Df Residuals:                          121   BIC:                             2877.
Df Model:                                9
Covariance Type:                 nonrobust
=====================================================================================================
                                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------------------------
Intercept                         498.0839   3276.307      0.152      0.879   -5988.231    6984.398
C(gdp_per_capita_bins)[T.2=20%]  2066.9124   4721.661      0.438      0.662   -7280.862    1.14e+04
C(gdp_per_capita_bins)[T.3=30%]  5716.7560   4721.661      1.211      0.228   -3631.018    1.51e+04
C(gdp_per_capita_bins)[T.4=40%]  1.067e+04   4721.661      2.259      0.026    1319.964       2e+04
C(gdp_per_capita_bins)[T.5=50%]  1.552e+04   4721.661      3.286      0.001    6168.164    2.49e+04
C(gdp_per_capita_bins)[T.6=60%]  1.393e+04   4721.661      2.951      0.004    4584.203    2.33e+04
C(gdp_per_capita_bins)[T.7=70%]  1.796e+04   4721.661      3.804      0.000    8614.838    2.73e+04
C(gdp_per_capita_bins)[T.8=80]   2.251e+04   4721.661      4.768      0.000    1.32e+04    3.19e+04
C(gdp_per_capita_bins)[T.9=90%]  2.053e+04   4721.661      4.349      0.000    1.12e+04    2.99e+04
C(gdp_per_capita_bins)[T.10=100%] 2.455e+04  4721.661      5.200      0.000    1.52e+04    3.39e+04
==============================================================================
Omnibus:                       15.796   Durbin-Watson:                   2.182
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               18.323
Skew:                           0.761   Prob(JB):                     0.000105
Kurtosis:                       4.020   Cond. No.                         10.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

The P-value is 1.82e-07 which is sufficiently lower than our significance limit and hence we reject H0. We have also run a post-hoc test to check that the difference between the means is still significant even after we check for type-1 errors. We can carry out post-hoc tests with the help of the multicomp module, utilizing a Tukey Honestly Significant Difference (Tukey HSD) test:

In [71]:
```
multi_comparison = multi.MultiComparison(anova_df["total_cases_per_million"], anova_df["gdp_per_capita_bins"])
results = multi_comparison.tukeyhsd()
print(results)
```

```
      Multiple Comparison of Means - Tukey HSD, FWER=0.05
==============================================================
 group1 group2   meandiff  p-adj     lower       upper    reject
--------------------------------------------------------------
10=100%  1=10%  -24552.408  0.001  -39772.688 -9332.1279   True
10=100%  2=20% -22485.4956  0.001 -37985.0702 -6985.9211   True
10=100%  3=30%  -18835.652 0.0056 -34335.2265 -3336.0775   True
10=100%  4=40% -13884.6703  0.121 -29384.2448  1614.9042  False
```

```
10=100%   5=50%  -9036.4704 0.6594 -24536.0449  6463.1042  False
10=100%   6=60%  -10620.431 0.4594 -26120.0055  4879.1435  False
10=100%   7=70%  -6589.7958    0.9 -22089.3704  8909.7787  False
10=100%    8=80   -2041.744    0.9 -17541.3185 13457.8305  False
10=100%   9=90%  -4017.7771    0.9 -19517.3516 11481.7975  False
  1=10%   2=20%   2066.9124    0.9 -13153.3677 17287.1924  False
  1=10%   3=30%    5716.756    0.9  -9503.5241  20937.036  False
  1=10%   4=40%  10667.7377 0.4263  -4552.5424 25888.0177  False
  1=10%   5=50%  15515.9376 0.0419    295.6576 30736.2177   True
  1=10%   6=60%   13931.977 0.1033  -1288.3031  29152.257  False
  1=10%   7=70%  17962.6121 0.0082   2742.3321 33182.8922   True
  1=10%    8=80   22510.664  0.001   7290.3839  37730.944   True
  1=10%   9=90%  20534.6309 0.0012   5314.3509  35754.911   True
  2=20%   3=30%   3649.8436    0.9 -11849.7309 19149.4182  False
  2=20%   4=40%   8600.8253 0.7138  -6898.7492 24100.3998  False
  2=20%   5=50%  13449.0252 0.1492  -2050.5493 28948.5998  False
  2=20%   6=60%  11865.0646 0.2969  -3634.5099 27364.6392  False
  2=20%   7=70%  15895.6998 0.0396    396.1252 31395.2743   True
  2=20%    8=80  20443.7516 0.0017   4944.1771 35943.3262   True
  2=20%   9=90%  18467.7185 0.0073    2968.144 33967.2931   True
  3=30%   4=40%   4950.9817    0.9 -10548.5928 20450.5562  False
  3=30%   5=50%   9799.1816 0.5641  -5700.3929 25298.7562  False
  3=30%   6=60%    8215.221  0.762  -7284.3535 23714.7955  False
  3=30%   7=70%  12245.8562 0.2549  -3253.7184 27745.4307  False
  3=30%    8=80   16793.908 0.0226   1294.3335 32293.4825   True
  3=30%   9=90%  14817.8749  0.074   -681.6996 30317.4495  False
  4=40%   5=50%   4848.1999    0.9 -10651.3746 20347.7745  False
  4=40%   6=60%   3264.2393    0.9 -12235.3352 18763.8138  False
  4=40%   7=70%   7294.8745  0.877  -8204.7001  22794.449  False
  4=40%    8=80  11842.9263 0.2995  -3656.6482 27342.5008  False
  4=40%   9=90%   9866.8932 0.5557  -5632.6813 25366.4678  False
  5=50%   6=60%  -1583.9606    0.9 -17083.5352 13915.6139  False
  5=50%   7=70%   2446.6745    0.9    -13052.9 17946.2491  False
  5=50%    8=80   6994.7264    0.9  -8504.8482 22494.3009  False
  5=50%   9=90%   5018.6933    0.9 -10480.8812 20518.2678  False
  6=60%   7=70%   4030.6352    0.9 -11468.9394 19530.2097  False
  6=60%    8=80    8578.687 0.7166  -6920.8875 24078.2615  False
  6=60%   9=90%   6602.6539    0.9  -8896.9206 22102.2285  False
  7=70%    8=80   4548.0518    0.9 -10951.5227 20047.6264  False
  7=70%   9=90%   2572.0188    0.9 -12927.5558 18071.5933  False
   8=80   9=90%  -1976.0331    0.9 -17475.6076 13523.5415  False
---------------------------------------------------------------
```

Now we have some better insight into which groups in our comparison have statistically significant differences.

If the reject column has a label of False, we know it's recommended that we reject the null hypothesis and assume that there is a significant difference between the two groups being compared.

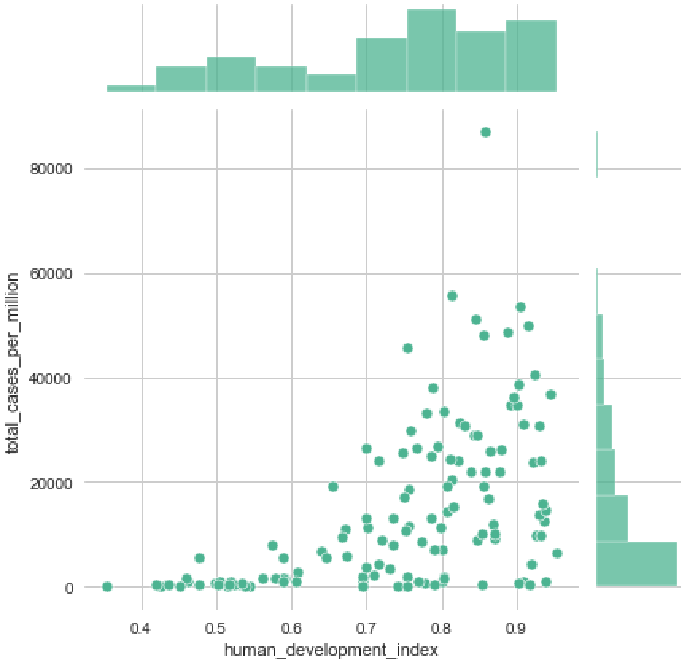
Hypothesis 2: Countries with higher human_development_index reported more number of cases of COVID

H0:There is no dependency of cases on Human development index of a country

H1:Countries with higher Human development index reported more number of cases of COVID

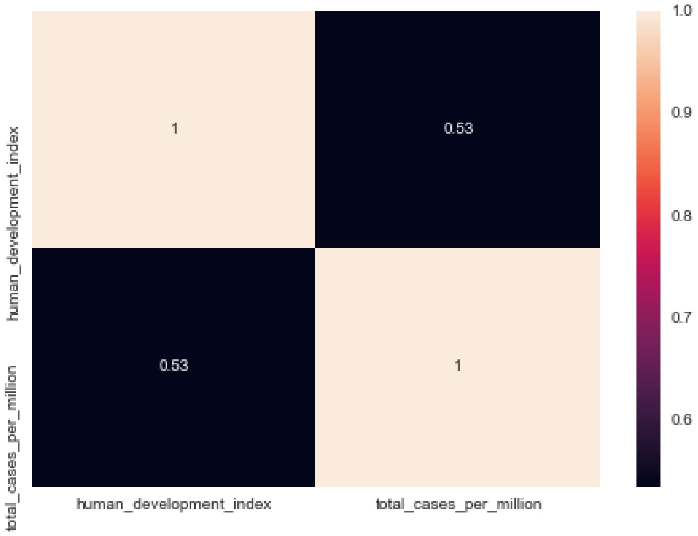
In [72]:
```
sns.jointplot(x=df_economy['human_development_index'], y=df_economy['total_cases_per_million'],  color="#4CB391
```

Out[72]: <seaborn.axisgrid.JointGrid at 0x2e8c5eb79d0>



In [73]:

```
correlation_mat = df_economy[['human_development_index','total_cases_per_million']].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [74]:

```
anova_df = df3[['human_development_index_bins','total_cases_per_million']].dropna()
anova = smf.ols(formula='total_cases_per_million ~ C(human_development_index_bins)', data=anova_df).fit()
print(anova.summary())
```

```
                              OLS Regression Results
===================================================================================
Dep. Variable:     total_cases_per_million   R-squared:                       0.365
Model:                                 OLS   Adj. R-squared:                  0.318
Method:                      Least Squares   F-statistic:                     7.788
Date:                     Fri, 11 Dec 2020   Prob (F-statistic):           5.28e-09
Time:                             12:50:17   Log-Likelihood:                -1432.3
No. Observations:                      132   AIC:                             2885.
Df Residuals:                          122   BIC:                             2913.
Df Model:                                9
Covariance Type:                 nonrobust
=======================================================================================================
==
                                              coef    std err          t      P>|t|      [0.025      0.97
5]
-------------------------------------------------------------------------------------------------------
--
Intercept                                 860.9435   3469.397      0.248      0.804   -6007.074     7728.9
61
C(human_development_index_bins)[T.2=20%]  713.8301   4906.468      0.145      0.885   -8999.014     1.04e+
04
C(human_development_index_bins)[T.3=30%] 4764.5518   5106.814      0.933      0.353   -5344.897     1.49e+
04
C(human_development_index_bins)[T.4=40%] 8447.3371   4999.933      1.689      0.094   -1450.530     1.83e+
04
```

```
C(human_development_index_bins)[T.5=50%]    1.432e+04    4999.933      2.864      0.005     4420.192     2.42e+
04
C(human_development_index_bins)[T.6=60%]    1.446e+04    4999.933      2.892      0.005     4560.141     2.44e+
04
C(human_development_index_bins)[T.7=70%]    2.523e+04    4999.933      5.045      0.000     1.53e+04     3.51e+
04
C(human_development_index_bins)[T.8=80]     2.213e+04    4999.933      4.427      0.000     1.22e+04      3.2e+
04
C(human_development_index_bins)[T.9=90%]    2.821e+04    4999.933      5.642      0.000     1.83e+04     3.81e+
04
C(human_development_index_bins)[T.10=100%]  1.662e+04    4906.468      3.387      0.001     6906.865     2.63e+
04
==============================================================================
Omnibus:                       43.012   Durbin-Watson:                   2.308
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              143.126
Skew:                           1.158   Prob(JB):                     8.33e-32
Kurtosis:                       7.545   Cond. No.                         10.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
```

The P-value is 5.28e-09 which is sufficiently lower than our significance limit and hence we reject H0. We have also run a post-hoc test to check that the difference between the means is still significant even after we check for type-1 errors. We can carry out post-hoc tests with the help of the multicomp module, utilizing a Tukey Honestly Significant Difference (Tukey HSD) test:

In [75]:

```
multi_comparison = multi.MultiComparison(anova_df["total_cases_per_million"], anova_df["human_development_index
results = multi_comparison.tukeyhsd()
print(results)
```

```
      Multiple Comparison of Means - Tukey HSD, FWER=0.05
===============================================================
 group1 group2   meandiff p-adj     lower      upper    reject
---------------------------------------------------------------
10=100%  1=10% -16619.7093 0.0311 -32433.2916   -806.127   True
10=100%  2=20% -15905.8792 0.0474 -31719.4615   -92.2969   True
10=100%  3=30% -11855.1575 0.3853 -28314.4558  4604.1408  False
10=100%  4=40%  -8172.3722 0.8065 -24287.1926  7942.4483  False
10=100%  5=50%  -2301.6501    0.9 -18416.4706 13813.1704  False
10=100%  6=60%  -2161.7008    0.9 -18276.5212 13953.1197  False
10=100%  7=70%   8606.1984 0.7544   -7508.622 24721.0189  False
10=100%   8=80   5512.5358    0.9 -10602.2847 21627.3562  False
10=100%  9=90%  11589.8495 0.3875  -4524.9709   27704.67  False
  1=10%  2=20%    713.8301    0.9 -15099.7522 16527.4124  False
  1=10%  3=30%   4764.5518    0.9 -11694.7466 21223.8501  False
  1=10%  4=40%   8447.3371 0.7735  -7667.4833 24562.1576  False
  1=10%  5=50%  14318.0592 0.1271  -1796.7613 30432.8797  False
  1=10%  6=60%  14458.0085 0.1197   -1656.812  30572.829  False
  1=10%  7=70%  25225.9077  0.001   9111.0873 41340.7282   True
  1=10%   8=80   22132.245  0.001   6017.4246 38247.0655   True
  1=10%  9=90%  28209.5588  0.001  12094.7383 44324.3793   True
  2=20%  3=30%   4050.7217    0.9 -12408.5766   20510.02  False
  2=20%  4=40%    7733.507 0.8593  -8381.3134 23848.3275  False
  2=20%  5=50%  13604.2291 0.1769  -2510.5913 29719.0496  False
  2=20%  6=60%  13744.1784 0.1667   -2370.642 29858.9989  False
  2=20%  7=70%  24512.0777  0.001   8397.2572 40626.8981   True
  2=20%   8=80   21418.415 0.0015   5303.5945 37533.2354   True
  2=20%  9=90%  27495.7287  0.001  11380.9083 43610.5492   True
  3=30%  4=40%   3682.7854    0.9 -13066.1416 20431.7123  False
  3=30%  5=50%   9553.5074  0.684  -7195.4195 26302.4344  False
  3=30%  6=60%   9693.4568 0.6678  -7055.4702 26442.3837  False
  3=30%  7=70%   20461.356 0.0052    3712.429 37210.2829   True
  3=30%   8=80  17367.6933 0.0355    618.7663 34116.6202   True
  3=30%  9=90%  23445.0071  0.001   6696.0801  40193.934   True
  4=40%  5=50%   5870.7221    0.9 -10539.8078  22281.252  False
  4=40%  6=60%   6010.6714    0.9 -10399.8585 22421.2013  False
  4=40%  7=70%  16778.5706 0.0407    368.0407 33189.1005   True
  4=40%   8=80  13684.9079 0.1903   -2725.622 30095.4378  False
  4=40%  9=90%  19762.2217 0.0063   3351.6918 36172.7516   True
  5=50%  6=60%    139.9493    0.9 -16270.5806 16550.4792  False
  5=50%  7=70%  10907.8485 0.5014  -5502.6814 27318.3784  False
  5=50%   8=80   7814.1858 0.8665  -8596.3441 24224.7157  False
  5=50%  9=90%  13891.4996 0.1746  -2519.0303 30302.0295  False
  6=60%  7=70%  10767.8992 0.5179  -5642.6307 27178.4291  False
  6=60%   8=80   7674.2365  0.883  -8736.2934 24084.7664  False
  6=60%  9=90%  13751.5503  0.185  -2658.9796 30162.0802  False
  7=70%   8=80  -3093.6627    0.9 -19504.1926 13316.8672  False
  7=70%  9=90%   2983.6511    0.9 -13426.8788  19394.181  False
   8=80  9=90%   6077.3138    0.9 -10333.2161 22487.8437  False
---------------------------------------------------------------
```

Now we have some better insight into which groups in our comparison have statistically significant differences.

If the reject column has a label of False, we know it's recommended that we reject the null hypothesis and assume that there is a significant difference between the two groups being compared.

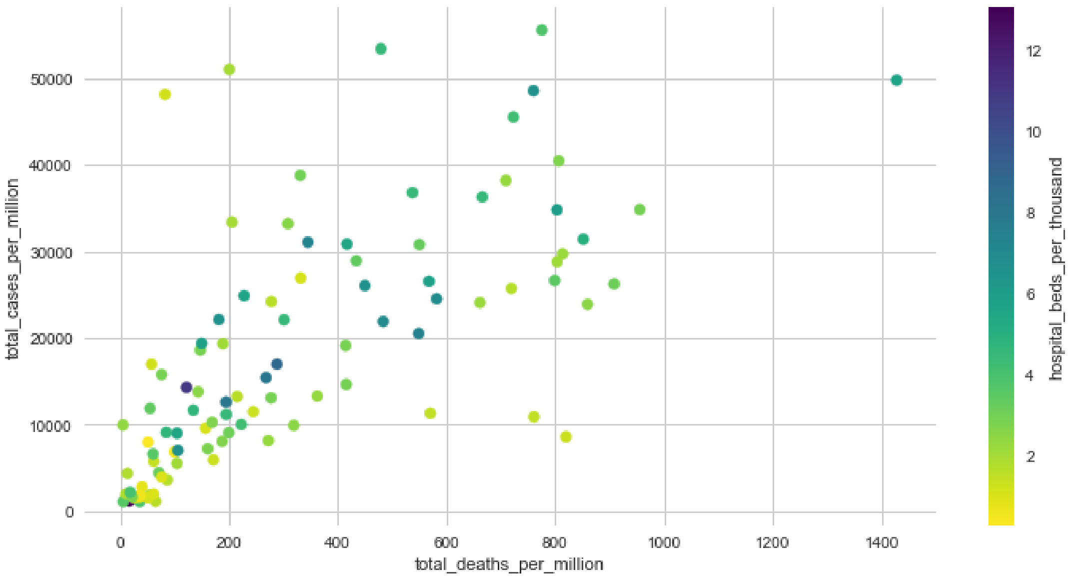
Effect of Hospital facilities upon the cases observed in covid

Hypothesis 1: hospital_beds_per_thousand in a countries with more than 1000 cases is directly proportionate to number of cases

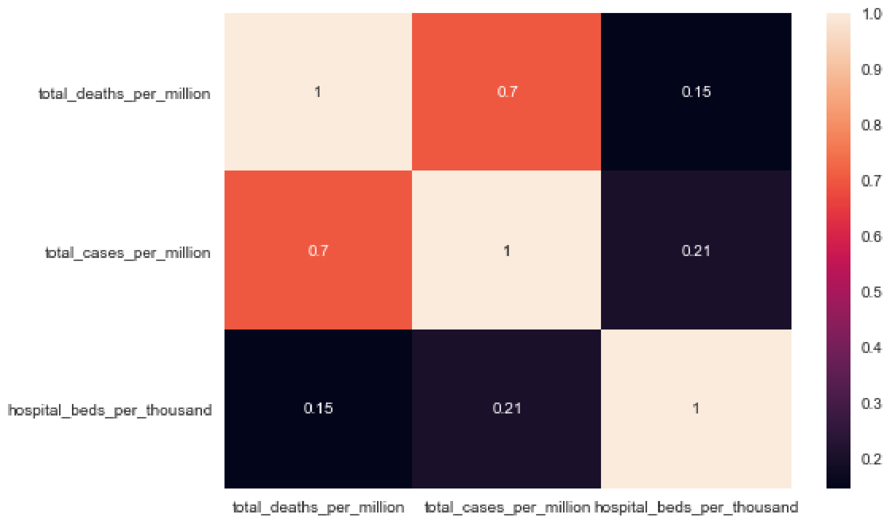
H0:hospital_beds_per_thousand in a countries with more than 1000 cases is not directly proportionate to number of cases
H1:hospital_beds_per_thousand in a countries with more than 1000 cases is directly proportionate to number of cases

In [76]:
```
df_temp = df_medical.loc[df_medical['total_cases_per_million']>1000]
```

In [77]:
```
plt.figure(figsize=[12,6])
plt.scatter(data=df_temp, x='total_deaths_per_million', y='total_cases_per_million', c='hospital_beds_per_thous
plt.colorbar(label='hospital_beds_per_thousand');
plt.xlabel('total_deaths_per_million');
plt.ylabel('total_cases_per_million');
axes = plt.gca()
```



In [78]:
```
correlation_mat = df_temp[['total_deaths_per_million','total_cases_per_million','hospital_beds_per_thousand']].
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [79]:
```
df_temp = df_temp.dropna(subset=['total_deaths_per_million','total_cases_per_million','hospital_beds_per_thousa
print("Assoc. - total_deaths_per_million and hospital_beds_per_thousand",pearsonr(df_temp['total_deaths_per_mil
print("Assoc. - total_cases_per_million and hospital_beds_per_thousand",pearsonr(df_temp['total_cases_per_milli
```

```
Assoc. - total_deaths_per_million and hospital_beds_per_thousand (0.14561547329408395, 0.15469208086346792)
Assoc. - total_cases_per_million and hospital_beds_per_thousand (0.20537675033470681, 0.043583827436443234)
```

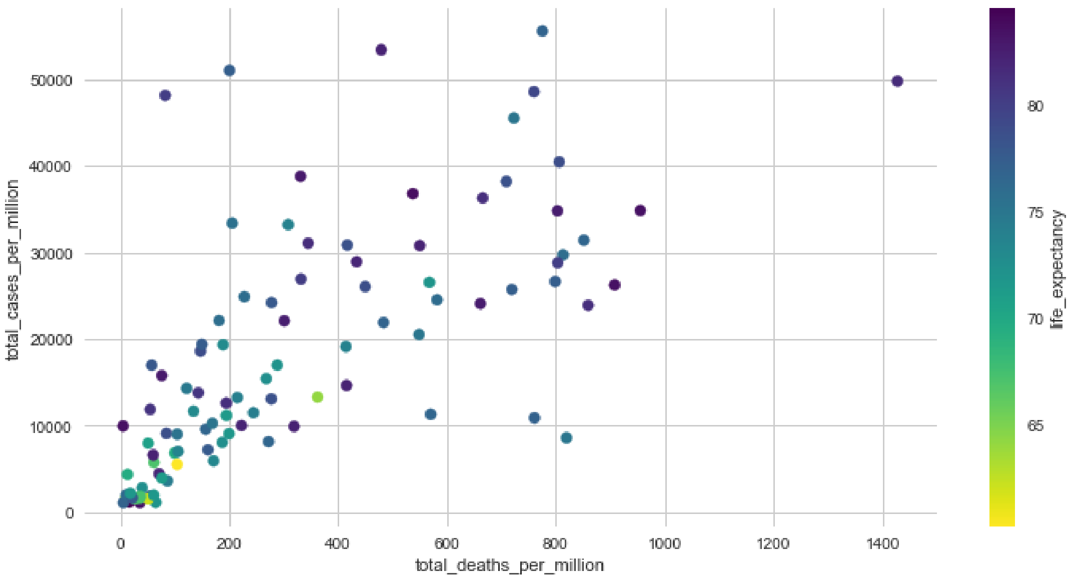
p value = 0.043583827436443234 < 0.05 so we reject H0 and accept that they are proportionate

But for total_deaths_per_million p value=0.15469208086346792 > 0.05 so here we cannot really say that hospital_beds_per_thousand in a countries with more than 1000 cases is directly proportionate to number of cases

Hypothesis 2: Countries with higher cases per million had higher life_expentancy

H0:Countries with higher cases per million had similar higher life_expentancy as countries with lower cases per million
H1:Countries with higher cases per million had higher life_expentancy

In [80]:
```
plt.figure(figsize=[12,6])
plt.scatter(data=df_temp, x='total_deaths_per_million', y='total_cases_per_million', c='life_expectancy', cmap=
plt.colorbar(label='life_expectancy');
plt.xlabel('total_deaths_per_million');
plt.ylabel('total_cases_per_million');
axes = plt.gca()
```



In [81]:
```
df_temp = df_temp.dropna(subset=['total_deaths_per_million','total_cases_per_million','life_expectancy'])
print("Assoc. - total_deaths_per_million and life_expectancy",pearsonr(df_temp['total_deaths_per_million'],df_t
print("Assoc. - total_cases_per_million and life_expectancy",pearsonr(df_temp['total_cases_per_million'],df_tem
```

```
Assoc. - total_deaths_per_million and life_expectancy (0.35857265583960357, 0.0003104102691965809)
Assoc. - total_cases_per_million and life_expectancy (0.4328552375764031, 9.506379634123818e-06)
```

for total_deaths_per_million and life expectancy <br. p value= 0.0003104102691965809 < 0.05
so we reject H0 and accept that Countries with higher cases per million had higher life_expentancy

Indian Economy

In this section we have found the relation between the number of cases in India verses the other factors related to economy. Those factors include:

- Inflation rate
- India infrastructure output
- Repo rates
- Consumer Spending

For this we have made a python code which works as follows:

- df_covid_monthly contains the information about India as we set the location to India.
- Next we create a list named data which contains other list such as total cases, inflation, infrastructure, repo rates, and consumer spending.
- Next we create a list for the months.
- After all the data we plot the histogram and scatter plot using the function plt.scatter and plt.hist.

Hypothesis 1: Inflation rate in India and world changed in 2019 and 2020 due to COVID

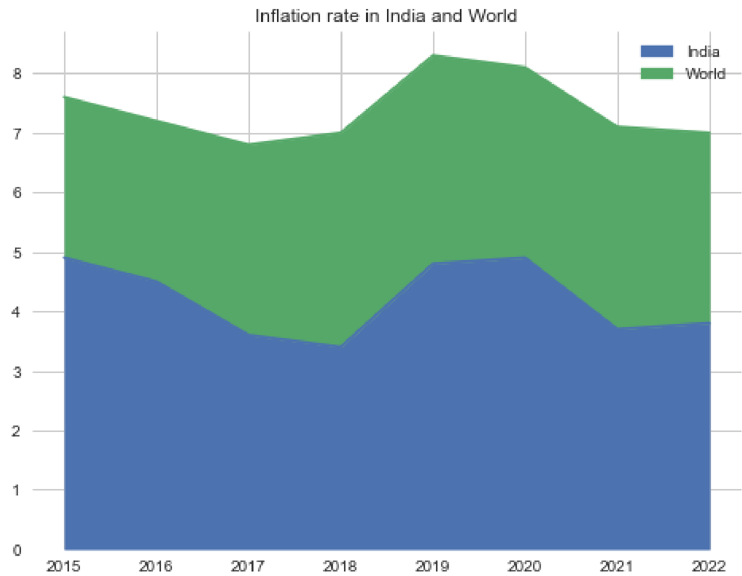
H0:Inflation rate in India and world remained same in 2019 and 2020
H1:Inflation rate in India and world changed in 2019 and 2020 due to COVID

In [82]:
```
df_IND_inf_modif = df_IND_inf.loc[:'World','2015':'2022']
df_IND_inf_modif = df_IND_inf_modif.T
df_IND_inf_modif = df_IND_inf_modif.drop(['Venezuela'], axis=1)
df_IND_inf_modif
```

Out[82]:

	Australia	Canada	China, People's Republic of	Ethiopia	France	Germany	India	Iraq	Japan	Kenya	United States	World
2015	1.5	1.1	1.4	9.6	0.1	0.7	4.9	1.4	0.8	6.6	0.1	2.7
2016	1.3	1.4	2.0	6.6	0.3	0.4	4.5	0.5	-0.1	6.3	1.3	2.7
2017	2.0	1.6	1.6	10.7	1.2	1.7	3.6	0.1	0.5	8.0	2.1	3.2
2018	1.9	2.3	2.1	13.8	2.1	2.0	3.4	0.4	1.0	4.7	2.4	3.6
2019	1.6	1.9	2.9	15.8	1.3	1.3	4.8	-0.2	0.5	5.2	1.8	3.5
2020	0.7	0.6	2.9	20.2	0.5	0.5	4.9	0.8	-0.1	5.3	1.5	3.2
2021	1.3	1.3	2.7	11.5	0.6	1.1	3.7	1.0	0.3	5.0	2.8	3.4
2022	1.5	1.6	2.6	8.0	1.0	1.3	3.8	1.5	0.7	5.0	2.1	3.2

In [83]:
```
lines = df_IND_inf_modif[['India', 'World']].plot(kind='area',title="Inflation rate in India and World",figsize
df_IND_inf_modif.index.name = 'year'
```



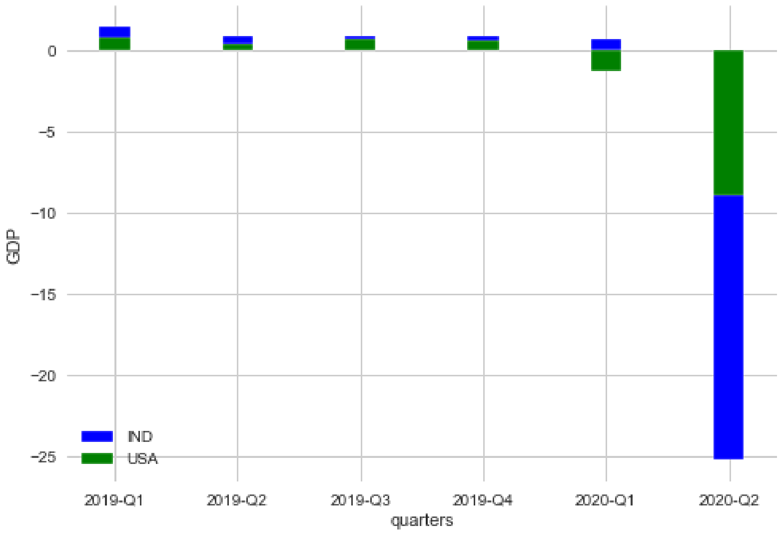
From the above graph it is clear that inflation rate has increased in India and decreased in thw world .So we reject H0

Hypothesis 2: GDP in India and USA fell in 2019-20 during the pandemic

H0:GDP in India and USA either remain unchanged or increased in 2019-20 during the pandemic
H1:GDP in India and USA fell in 2019-20 during the pandemic

In [84]:
```
column = ws_economy['A']
countries=['USA','IND']
time=['2019-Q1','2019-Q2','2019-Q3','2019-Q4','2020-Q1','2020-Q2']
values=ws_economy['G']

gdpind=[]
gdpusa=[]
for j in range(1,len(column)):
    if column[j].value=='USA':
        gdpusa.append(values[j].value)
    if column[j].value=='IND':
        gdpind.append(values[j].value)


gdpusa.pop()
data=[gdpind,gdpusa]
fig, ax = plt.subplots()
ax.bar(time, data[0], color = 'b', width = 0.25)
ax.bar(time, data[1], color = 'g', width = 0.25)
ax.legend(['IND','USA'],loc="lower left")
plt.xlabel('quarters')
```

```
plt.ylabel('GDP')
plt.show()
```

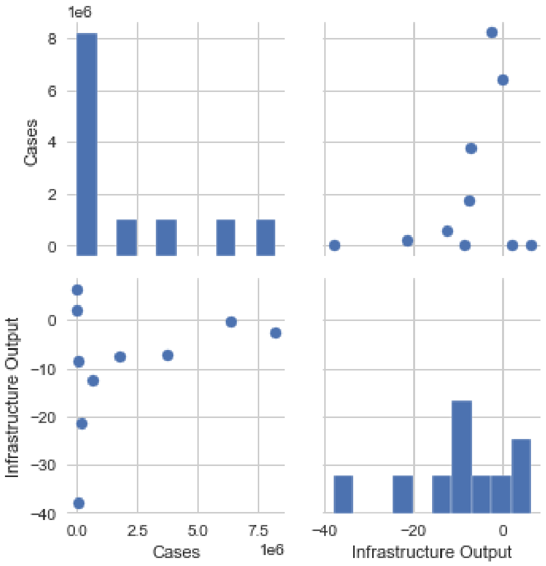


From the above plot we can infer that GDP of both India and USA fell in the pandemic period

Hypothesis 3: India's INFRASTRUCTURE OUTPUT is correlated with the pandemic

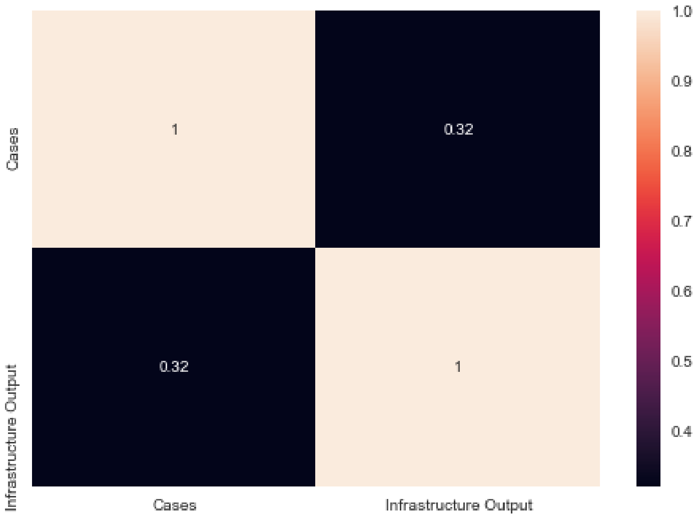
H0:India's infrastructure output is not correlated with the pandemic
H1:India's infrastructure output is correlated with the pandemic

In [85]:
```
df_covid_monthly = df_covid_data[df_covid_data["date"].str.endswith("01", na=False)]
df_covid_monthly = df_covid_monthly[df_covid_monthly["location"]=="India"]
data = [df_covid_monthly["total_cases"].values.tolist(),
        economyGetter("INDIA INFRASTRUCTURE OUTPUT"),
       ]
months = ["Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct"]
cols = ["Cases","Infrastructure Output"]
df_econInd_modif = pd.DataFrame(data, index = cols, columns = months)
df_econInd_modif = df_econInd_modif.T
g=sb.PairGrid(data=df_econInd_modif)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



Next we have made the correlation matrix for the factors mentioned above. In this matrix if the values are above 0 then they are directly proportional. And if the values are less than 0 then they are inversely proportional. In the matrix we can see that all the diagonal elements are having value 1 as they are relation to each other only If the value is equal to 0 then it indicates that there is no relation between the two elements being compared.

In [86]:
```
correlation_mat = df_econInd_modif.corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [87]:
```
df_temp = df_econInd_modif.dropna(subset=cols)
print("Assoc. - Cases and Infrastructure Output",pearsonr(df_temp['Cases'],df_temp['Infrastructure Output']))
```

```
Assoc. - Cases and Infrastructure Output (0.3199516817101196, 0.36745979672702955)
```

p value = 0.36745979672702955 > 0.05

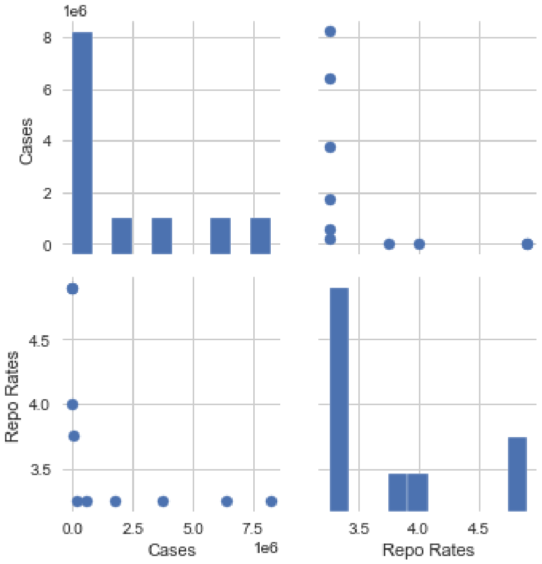
Hence we fail to reject H0

Hypothesis 4: India's REPO RATES is correlated with the pandemic

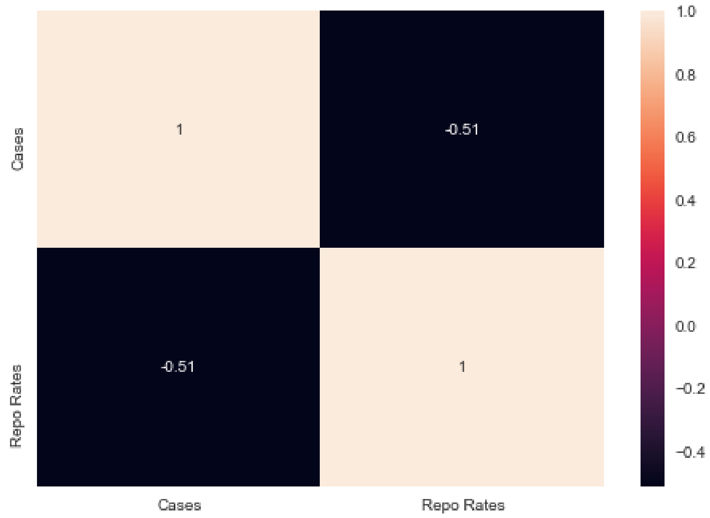
H0:India's REPO RATES is not correlated with the pandemic

H1:India's REPO RATES is correlated with the pandemic

In [88]:
```
df_covid_monthly = df_covid_data[df_covid_data["date"].str.endswith("01", na=False)]
df_covid_monthly = df_covid_monthly[df_covid_monthly["location"]=="India"]
data = [df_covid_monthly["total_cases"].values.tolist(),
        economyGetter("REPO RATES"),
        ]
months = ["Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct"]
cols = ["Cases","Repo Rates"]
df_econInd_modif = pd.DataFrame(data, index = cols, columns = months)
df_econInd_modif = df_econInd_modif.T
g=sb.PairGrid(data=df_econInd_modif)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [89]:
```
correlation_mat = df_econInd_modif.corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [90]:

```
print("Assoc. - Cases and Repo Rates",pearsonr(df_econInd_modif['Cases'],df_econInd_modif['Repo Rates']))
```

```
Assoc. - Cases and Repo Rates (-0.5134539647138775, 0.12903029666468196)
```

p value obtained is 0.12903029666468196 > 0.05

Hence we fail to reject H0

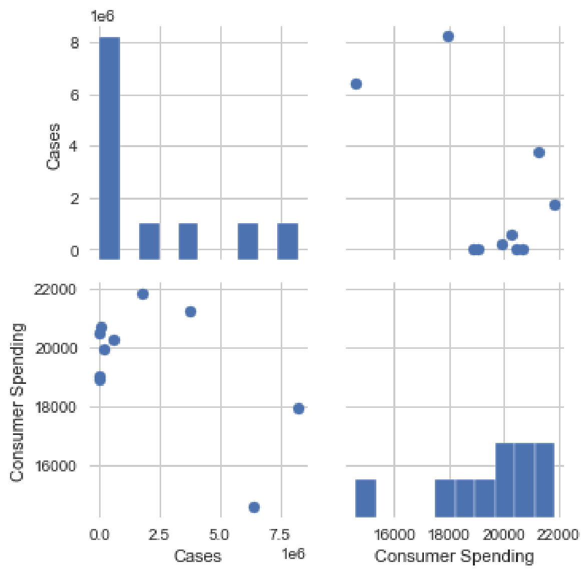
Hypothesis 5: India's CONSUMER SPENDING is correlated with the pandemic

H0: India's consumer spending is has zero correlation with the pandemic

H1: India's consumer spending is correlated with the pandemic

In [91]:

```
df_covid_monthly = df_covid_data[df_covid_data["date"].str.endswith("01", na=False)]
df_covid_monthly = df_covid_monthly[df_covid_monthly["location"]=="India"]
data = [df_covid_monthly["total_cases"].values.tolist(),
        economyGetter("CONSUMER SPENDING"),
       ]
months = ["Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sep","Oct"]
cols = ["Cases","Consumer Spending"]
df_econInd_modif = pd.DataFrame(data, index = cols, columns = months)
df_econInd_modif = df_econInd_modif.T
g=sb.PairGrid(data=df_econInd_modif)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [92]:

```
correlation_mat = df_econInd_modif.corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```

In [93]:

```
print("Assoc. - Cases and Consumer Spending",pearsonr(df_econInd_modif['Cases'],df_econInd_modif['Consumer Spen
```

```
Assoc. - Cases and Consumer Spending (-0.577750290560148, 0.08025723424337937)
```

p value obtained is 0.08025723424337937 > 0.05

Hence we fail to reject H0

Some important outcomes which we can look into by observing the correlation matrix:

i) We can see that the number of cases is directly proportional to inflation which means that if the number of cases rises then the price of the good increases.

ii) We also see that as the number of cases increases the infrastructural output increases.

iii) We observe that as the number of cases increases the repo rate or the rate at which commercial banks borrow money by selling their securities to the Central bank of our country decreases i.e. they are inversely proportional to each other.

iv) Lastly we observe that as the number of cases increases the consumer spending decreases as they are inversely proportional to each other.

From the observation we can take out the inference that:

i) As the cases increased people stopped coming out of their houses and because of this the consumption of goods decreased which led sellers to increase the price of the goods.

This was done by the sellers because the number of goods getting sold were very less and thus they wanted atleast they should make ample amout of money to survive hence increasing the price of the good and contributing to inflation.

ii) If we observe this output carefully month by month we observe that, for the months January, February and March the production was high as near to normal and the same goes for the months May and after May.

The main effect of the covid was observed in the month of April when there was complete shut down and no output was produced by any industries.

Thus the output is directly proportional because if we consider for the overall year the industrial output did not decreased.

iii) The reason for the decrease in the repo rates is hidden in the meaning of the repo rate.

The decrease in repo rates is to aim at bringing in growth and improving economic development in the country.

Consumers will borrow more from banks thus stabilizing the inflation.

A decline in the repo rate can lead to the banks bringing down their lending rate.

And as the bank bring down their lending rate people can borrow money from the banks at affordable rate of interest. Thus because of this the repo rates have decreased as the number of cases increased.

iv) This is quite obvious that the consumer spending will decrease as the cases rises because less people will be going out of their houses when cases rises thus they will spend less in the market.

Another reason is that because of this situation many have lost their jobs and thus to survive in this situation people have dropped their standard of living and are spending less in the market thus decreasing the consumer spending.

So in conclusion we can say that Economy is drastically affected by rise in the cases of Covid. Null hypothesis is rejected, and we accept alternate hypothesis

Effect of other diseases upon the cases and fatalities of Covid across several countries

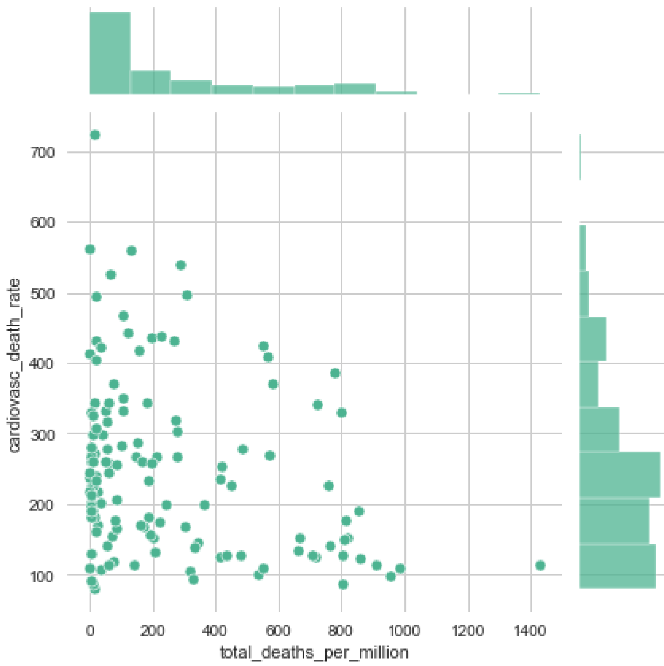
To find the relation between the factors that constitute diseases such as cardiovascular_deathrate, female_smokers, male_smokers and diabetes_prevalence. For this we have taken 5 constituents on Y-axis and the same on the X-axis. These

are:
i) cardiovascular_deathrate
ii) female_smokers
iii) male_smokers
iv) total_deaths_per_million
v) diabetes_prevalence
There are 25 plots out of which five are histograms and twenty are scatter plot. These plots are taken for 100 countries and average of all are considered.

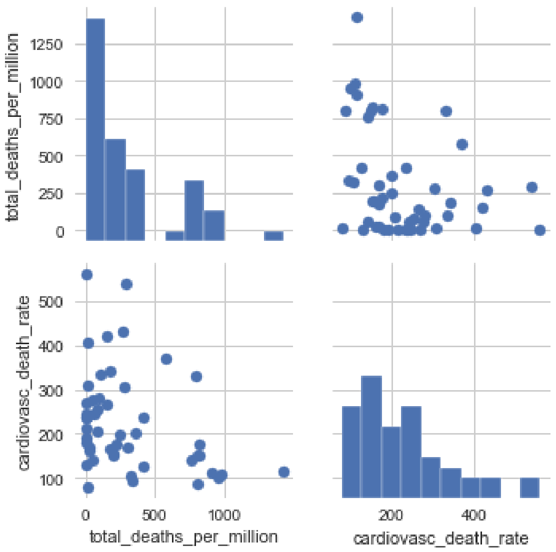
Hypothesis 1: cardiovascular_death_rate is correlated with the pandemic fatalities

H0:cardiovascular_death_rate is not correlated with the pandemic fatalities
H1:cardiovascular_death_rate is correlated with the pandemic fatalities

In [94]:
```
sns.jointplot(x=df_disease['total_deaths_per_million'], y=df_disease['cardiovasc_death_rate'],  color="#4CB391"
disease_list = ['total_deaths_per_million','cardiovasc_death_rate']
```



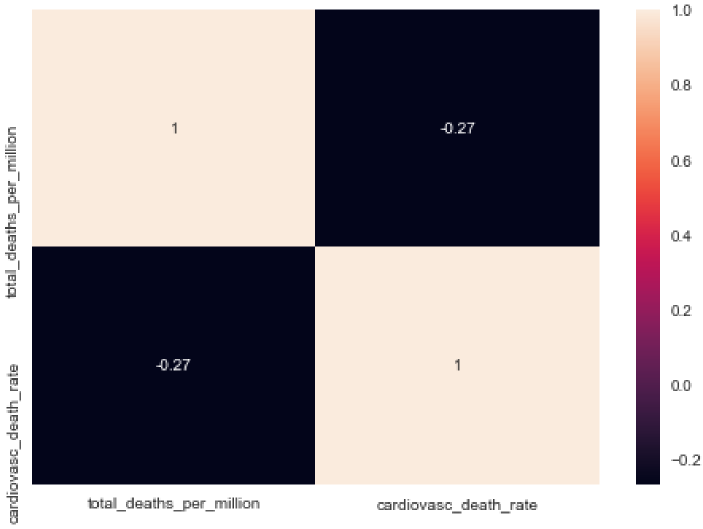
In [95]:
```
g=sb.PairGrid(data=df_disease.sample(n=50), vars=disease_list)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



Next we have made the correlation matrix for the factors mentioned above. In this matrix if the values are above 0 then they are directly proportional. And if the values are less than 0 then they are inversely proportional. In the matrix we can see that all the diagonal elements are having value 1 as they are relation to each other only If the value is equal to 0 then it indicates that there is no relation between the two elements being compared.

Correlation Matrix as a Heat map of the above plot

In [96]:
```
correlation_mat = df_disease[disease_list].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [97]:
```
print("Assoc. - total_deaths_per_million and cardiovasc_death_rate",pearsonr(df_disease['total_deaths_per_milli
```

```
Assoc. - total_deaths_per_million and cardiovasc_death_rate (-0.26659252957931284, 0.0019234133327449881)
```

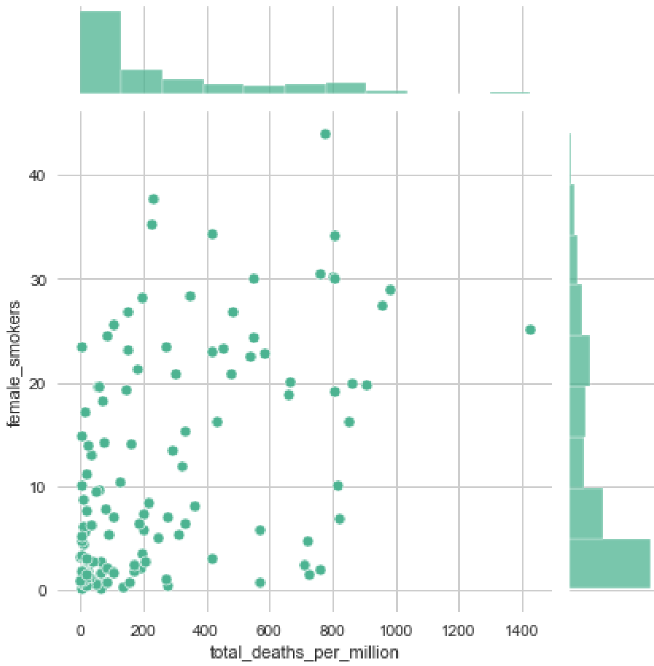
p value obtained is .0019234133327449881 <0.05 so we reject H0

Hypothesis 2: female_smokers proportion in a country is correlated with the pandemic fatalities

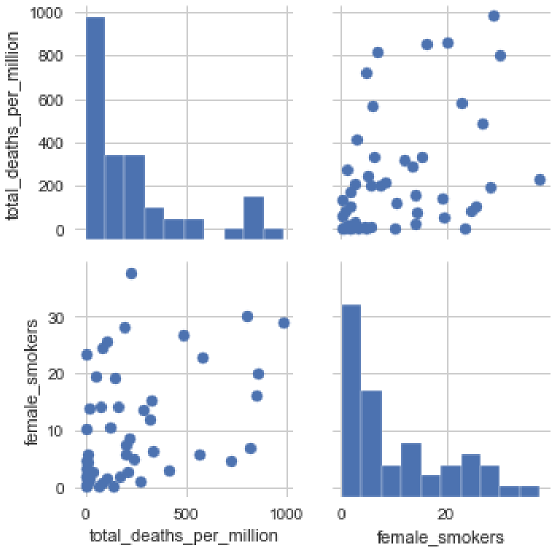
H0:female_smokers proportion in a country is not correlated with the pandemic fatalities

H1:female_smokers proportion in a country is correlated with the pandemic fatalities

In [98]:
```
sns.jointplot(x=df_disease['total_deaths_per_million'], y=df_disease['female_smokers'],  color="#4CB391")
disease_list = ['total_deaths_per_million','female_smokers']
```

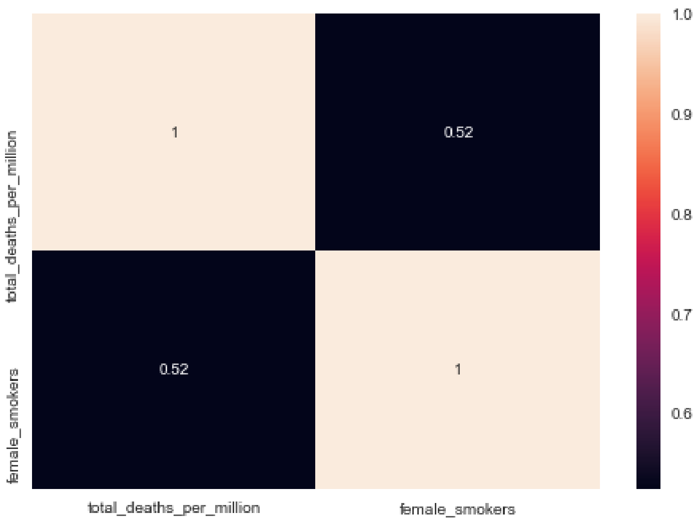


In [99]:
```
g=sb.PairGrid(data=df_disease.sample(n=50), vars=disease_list)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [100…

```
correlation_mat = df_disease[disease_list].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [101…

```
print("Assoc. - total_deaths_per_million and female_smokers",pearsonr(df_disease['total_deaths_per_million'],df
```

```
Assoc. - total_deaths_per_million and female_smokers (0.5239179743698653, 9.707526273421715e-11)
```

p value obtained is 9.707526273421715e-11 < 0.05

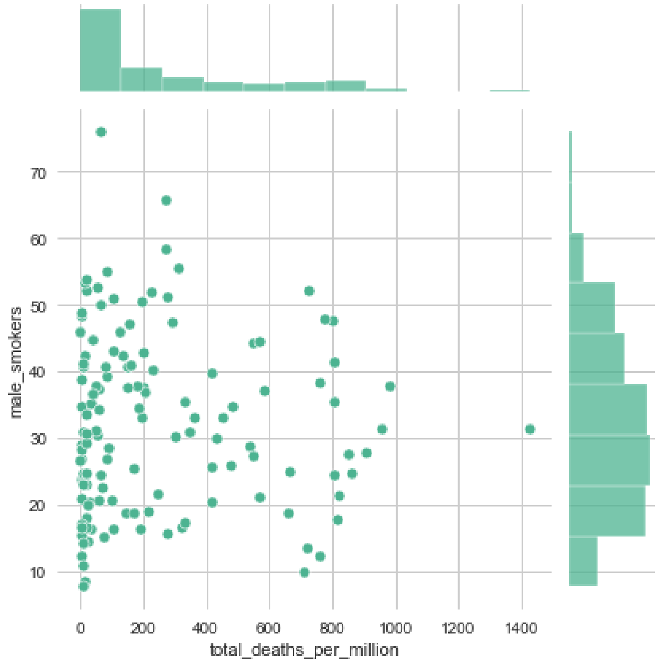
Hence we reject H0 and conclude female_smokers proportion in a country is correlated with the pandemic fatalities

Hypothesis 3: male_smokers proportion in a country is correlated with the pandemic fatalities

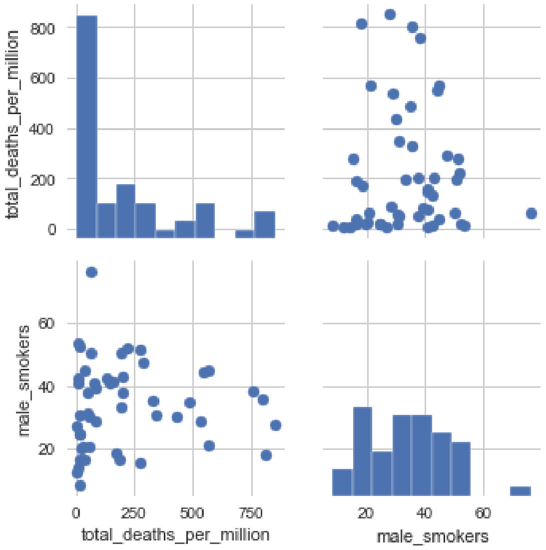
H0: male_smokers proportion in a country is not correlated with the pandemic fatalities H1: male_smokers proportion in a country is correlated with the pandemic fatalities

In [102…

```
sns.jointplot(x=df_disease['total_deaths_per_million'], y=df_disease['male_smokers'],  color="#4CB391")
disease_list = ['total_deaths_per_million','male_smokers']
```



In [103…
```
g=sb.PairGrid(data=df_disease.sample(n=50), vars=disease_list)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [104…
```
correlation_mat = df_disease[disease_list].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```

In [105…
```
print("Assoc. - total_deaths_per_million and male_smokers",pearsonr(df_disease['total_deaths_per_million'],df_d
```

```
Assoc. - total_deaths_per_million and male_smokers (-0.016923705180515742, 0.8466886204386115)
```

p value obtained is 0.8466886204386115 > 0.05

So we fail to reject H0.

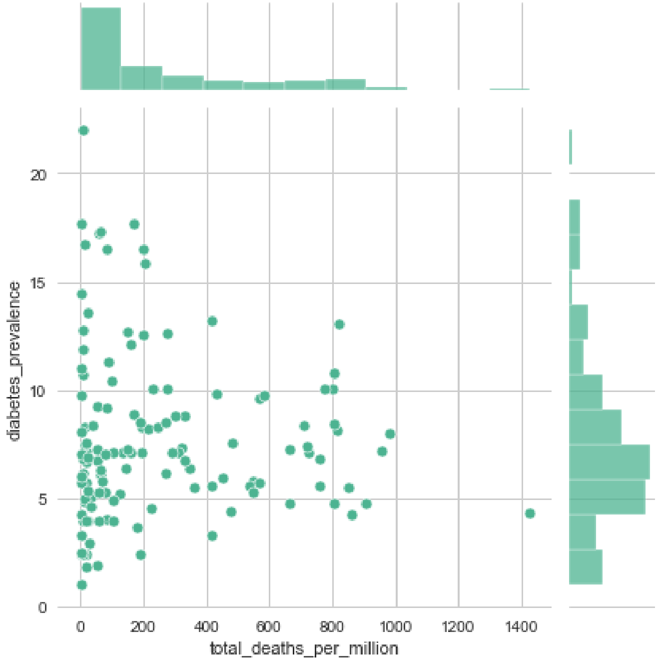
Hypothesis 4: diabetes_prevalence in a country is correlated with the pandemic fatalities

H0:diabetes_prevalence in a country is not correlated with the pandemic fatalities

H1:diabetes_prevalence in a country is correlated with the pandemic fatalities

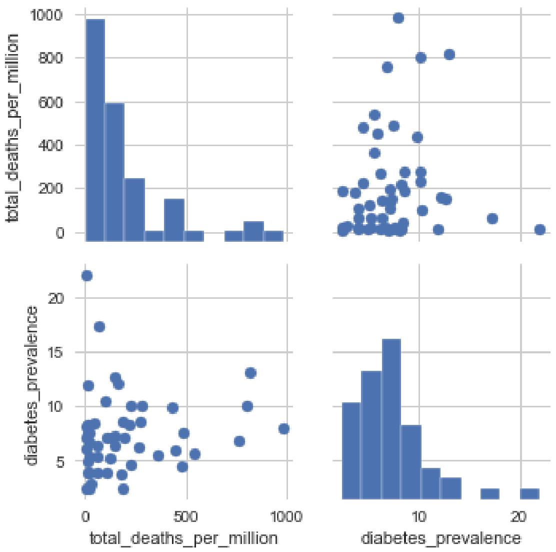
In [106…

```
sns.jointplot(x=df_disease['total_deaths_per_million'], y=df_disease['diabetes_prevalence'],  color="#4CB391")
disease_list = ['total_deaths_per_million','diabetes_prevalence']
```



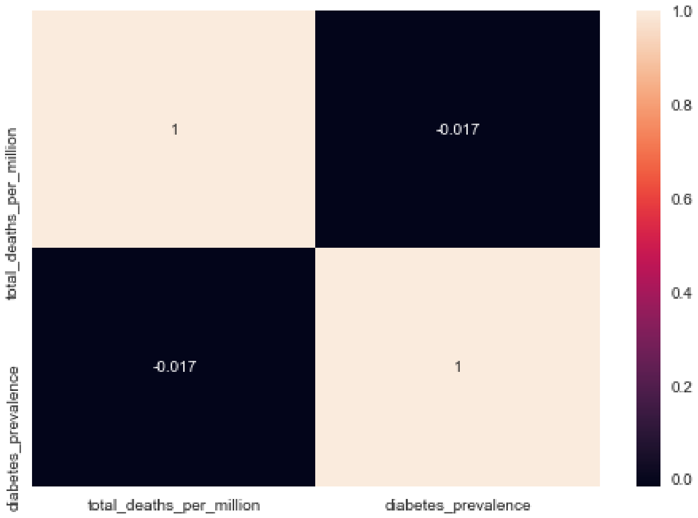
In [107…

```
g=sb.PairGrid(data=df_disease.sample(n=50), vars=disease_list)
g.map_offdiag(plt.scatter)
g.map_diag(plt.hist);
```



In [108…

```
correlation_mat = df_disease[disease_list].corr()
sns.heatmap(correlation_mat, annot = True)
plt.show()
```



In [109…

```
print("Assoc. - total_deaths_per_million and diabetes_prevalence",pearsonr(df_disease['total_deaths_per_million
```

```
Assoc. - total_deaths_per_million and diabetes_prevalence (-0.016500735864206l1, 0.8504750560492359)
```

p value= 0.8504750560492359 > 0.05
So we fail to reject H0

From the grpahs it is not very clear how the diabetes_prevelance,female_smokers,male_smokers are correlated to the mortality rate But we can get the analyse the correlation from the associated correlation matrix. There is a positive correlation between percentage of female smokers in country and total deaths per million. Which means the higher the female smokers in a country , more people die of having contracted the virus. There is a weak negative correlation (-0.017) in the data above between diabetes prevelance and fatal cases per population. The higher the percentage of a countrys population with diabetes, the lower the amount of fatal cases per population. Also there is a week negative correlation between male smokers and fatal cases per population. The higher the percentage of a countrys population with male smokers, the lower the amount of fatal cases per population. These three ideas are counterintuitive

Contributions:

Collection of Data : Pruthvi Raj DJ,Akshat,Siddharth,Ananya

Analysis of collected data: Everyone (discussed in meetings)

data extraction: Siddharth, Ashish

finalizing plots to make: Ashish and Atul

Coding and plotting: Ashish and Atul

Hypothesis making: Everyone (discussed in meetings)

Interpretation of hypothesis: Ananya, (economy : Akshat)

Mathematical support: Ananya

Compiling and organizing: Atul and Ashish

Documentation: Siddharth, Akshat, Atul

However, the contributions were fairly divided and everyone worked enthusiastically. The above list is only a summary of major contributers of each domain