CS-2375 Project: Covid-19 Data Visualization Portal

Manya Sachdev, Samveg Bhansali, Aaryann Mavani

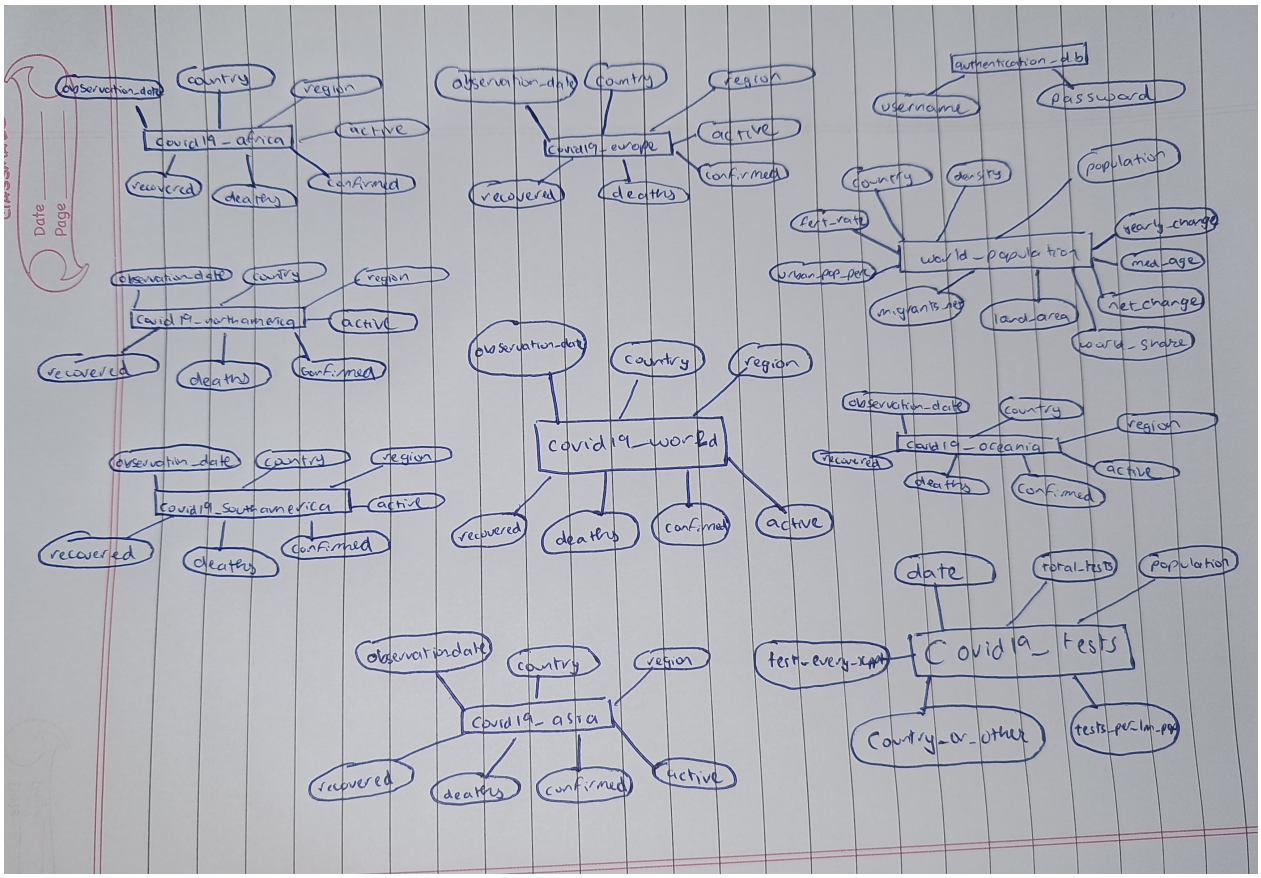
Report Due date: October 29, 2023, 11:59pm IST

Demo video URL:

`https://drive.google.com/file/d/1ocoWBhS_J09HUMOKFm8Y8VkAluy_CzMJ/view`

1 Section 1

The global impact of the Covid-19 pandemic highlighted the need for accurate and easily accessible information. However, the internet was flooded with unreliable data, making it difficult to obtain credible insights. Our Data Visualization Portal aims to solve this problem by providing a simple and clear interface that presents complex data in an easy-to-understand format. This portal will allow authorized users to regularly update the information, ensuring that it remains reliable and up-to-date. Our main goal is to create a strong system that not only addresses the issue of misinformation but also helps users effectively navigate complex public health data.



authentication_db	username, password
covid19_asia	observation_date, country, region, confirmed, deaths, recovered, active
covid19_europe	observation_date, country, region, confirmed, deaths, recovered, active
world_population	country, population, yearly_change, net_change, density, land_area, migrants_net, fert_rate
covid19_oceania	observation_date, country, region, confirmed, deaths, recovered, active
covid19_africa	observation_date, country, region, confirmed, deaths, recovered, active
covid19_northamerica	observation_date, country, region, confirmed, deaths, recovered, active
covid19_tests	date, country_or_other, total_tests, population, tests_per_1m_pop, test_every_x_x_ppl
covid19_southamerica	observation_date, country, region, confirmed, deaths, recovered, active
covid19_world	observation_date, country, region, confirmed, deaths, recovered, active

2 Section 2

1. Sources - John Hopkins University (https://github.com/CSSEGISandData/COVID-19)
   Worldometer (https://www.worldometers.info/coronavirus/)

2. We downloaded the dataset from a Kaggle where a contributor had collated the data.
   (https://www.kaggle.com/datasets/okwirjulius/covid19-cases-in-africa?select=covid19_africa.csv)

3. Cleanup steps involved dealing with the NA values appropriately for columns with varchar and integer values in the instances when the data is not available.

4. Statistics

Table	Number of Tuples	time to load	raw datasize	clean datasize
authentication_db	1	0.16ms	8Kb	8K
world_population	235	1.1ms	72Kb	72Kb
covid19_tests	44613	7138ms	4.9Mb	4.6Mb
covid19_asia	42628	6820ms	5.8Mb	5.6Mb
covid19_europe	63361	10137.76ms	9.7Mb	9.3Mb
covid19_oceania	3469	555.04ms	731Kb	608Kb
covid19_africa	15454	2472.64ms	2.9Mb	2.8Mb
covid19_northamerica	21015	3362.4ms	2.9Mb	2.7Mb
covid19_southamerica	38282	6124.48ms	5.6Mb	5.4Mb
covid19_world	185144	29623.04ms	27.6Mb	26Mb

3 Section 3

User's view of the system (this should be an itemized list)

1. The platform provides users with comprehensive country-specific cumulative statistics. By specifying a particular date range, users can access continent-level data for the selected period. Additionally, administrators can log in to the system and efficiently update data across various tables, ensuring the accuracy and relevance of the information presented.

2. - We built indexes on the date, and country columns of the world and continent data tables. We created a joined view with all columns from world_population and covid19_tests table. We have a functionality to allow admin to add data to the database. Finally, we created a trigger to ensure that every time covid19_world table is updated, the logged in user is verified as the admin.
   - a. Continent wise statistics for a particular date-
     SELECT observation_date, SUM(confirmed), SUM(deaths), SUM(recovered), SUM(active) FROM tablename GROUP BY observation_date

     b. Country wise statistics for a particular date-
     SELECT observation_date, country, SUM(confirmed), SUM(deaths), SUM(recovered), SUM(active) FROM covid19_world WHERE country = %s AND observation_date = %s GROUP BY observation_date, country", (country_name, specified_date)

     c. Statistics between a particular date range-
     SELECT observation_date, SUM(confirmed), SUM(deaths), SUM(recovered), SUM(active) FROM

tablename WHERE observation_date BETWEEN %s AND %s GROUP BY observation_date", (date1, date2)

d. Fetch World Population Statistics-
SELECT * FROM world_population

c. Fetch total testing data by date
SELECT date, SUM(total_tests) FROM covid19_tests GROUP BY date

e. Update data in the continent tables-
"""INSERT INTO table_name (observation_date, country, region, confirmed, deaths, recovered, active) VALUES (%s, %s, %s, %s, %s, %s, %s) ON CONFLICT (observation_date, country, region) DO NOTHING """, (date, country, region, confirmed, deaths, recovered, active))

f. Update data in the world table-
"""INSERT INTO covid19_world (observation_date, country, region, confirmed, deaths, recovered, active) VALUES (%s, %s, %s, %s, %s, %s, %s) ON CONFLICT (observation_date, country, region) DO NOTHING """, (date, country, region, confirmed, deaths, recovered, active)

g. Join data from continents into one table-
"""SELECT observation_date, SUM(confirmed) as total_confirmed, SUM(deaths) as total_deaths, SUM(recovered) as total_recovered, SUM(active) as total_active FROM (SELECT observation_date, confirmed, deaths, recovered, active FROM covid19_africa UNION ALL SELECT observation_date, confirmed, deaths, recovered, active FROM covid19_asia UNION ALL SELECT observation_date, confirmed, deaths, recovered, active FROM covid19_europe UNION ALL SELECT observation_date, confirmed, deaths, recovered, active FROM covid19_north_america UNION ALL SELECT observation_date, confirmed, deaths, recovered, active FROM covid19_south_america) as all_continents GROUP BY observation_date; """)

h. Update data in the tests table-
"""INSERT INTO covid19_tests (date, country_or_other, total_tests, population, tests_per_1m_pop, test_every_x_ppl) VALUES (%s, %s, %s, %s, %s, %s) ON CONFLICT (date, country_or_other) DO NOTHING """, (date, country, total_test, population, tests_per_million, test_per_person))

3. List of queries and run times.

Query Number	Average Running Time
1	371ms
2	1231ms
4	727ms
5	1796ms
6	923ms
7	1498ms
8	2337ms
9	1023ms