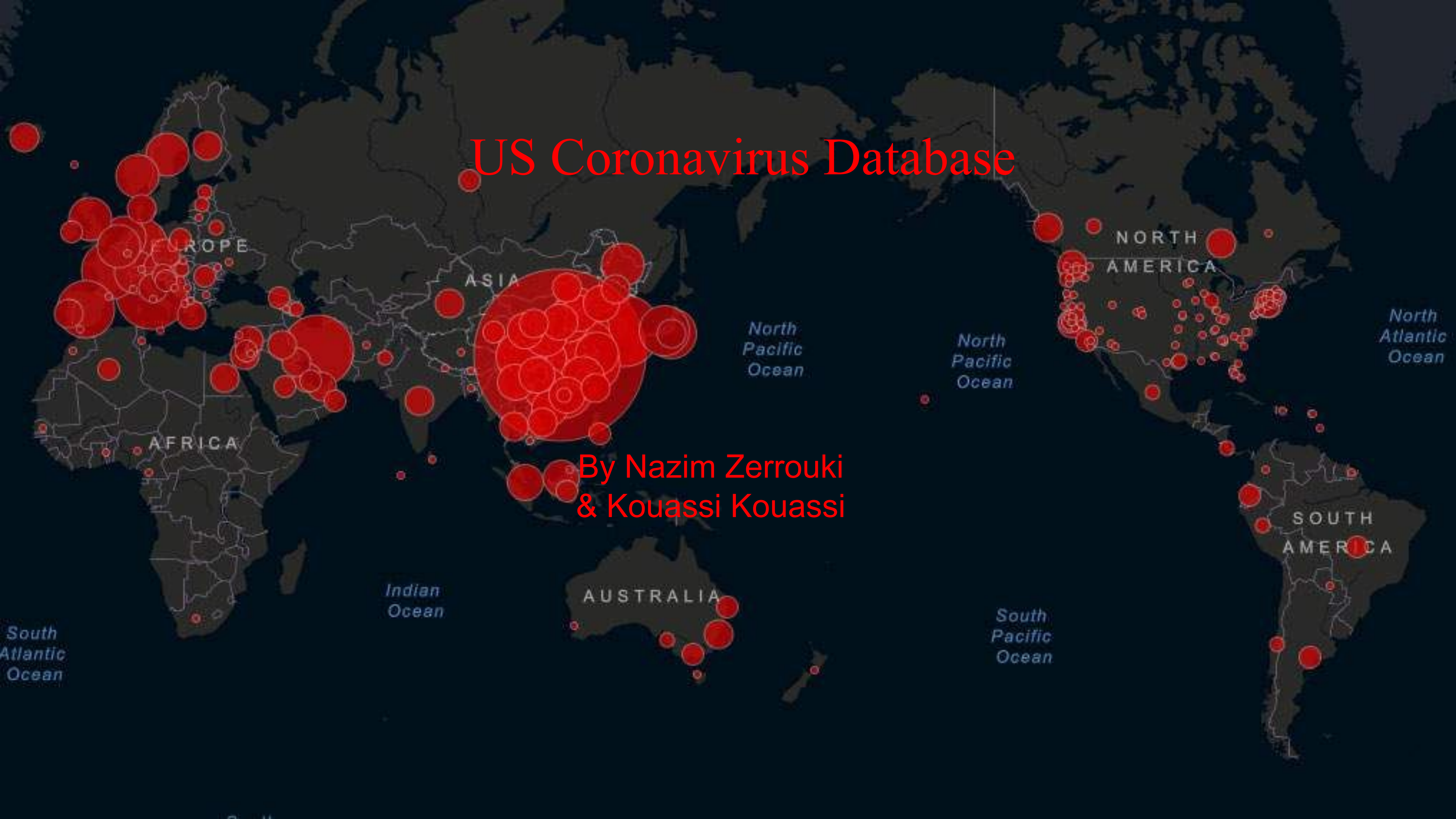
US Coronavirus Database

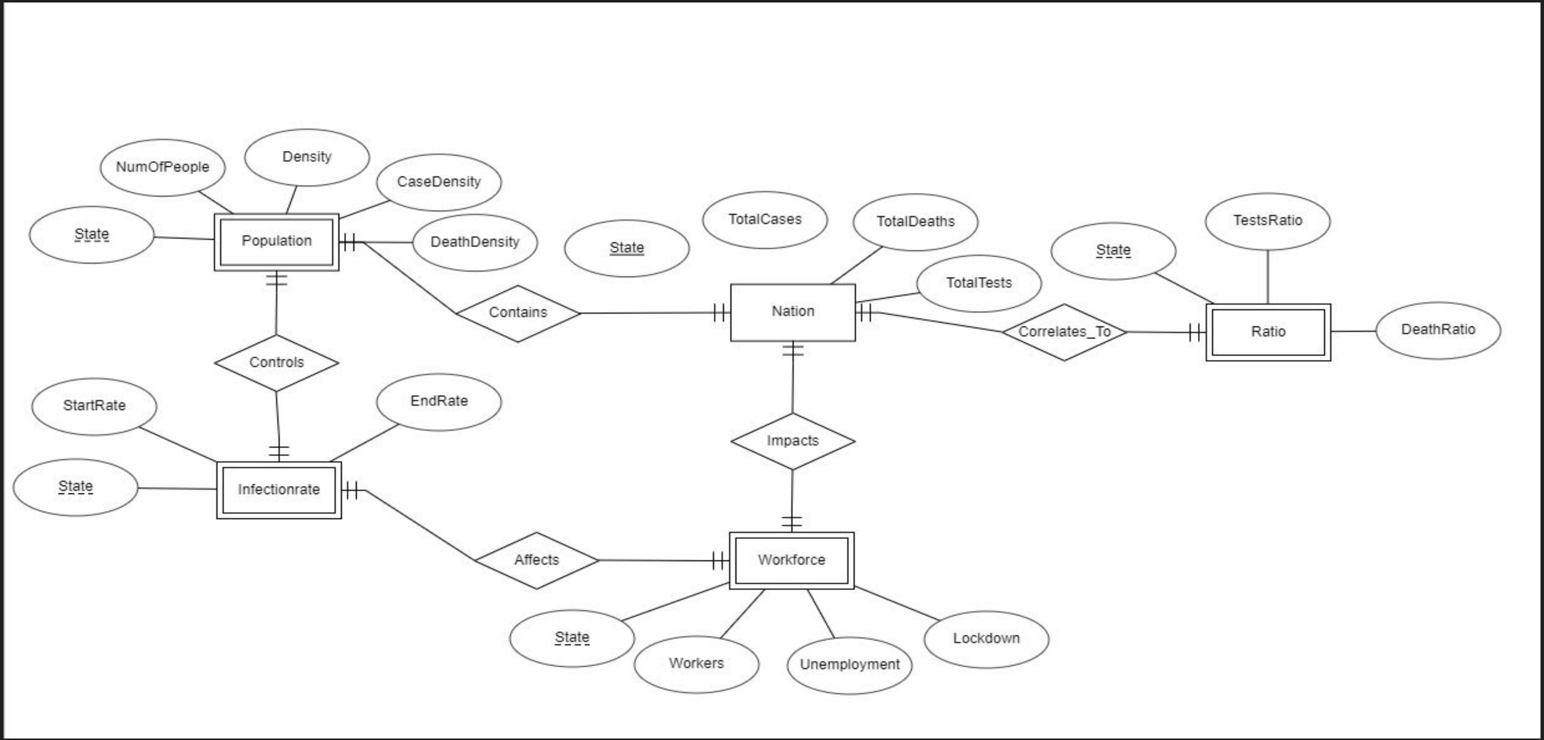


By Nazim Zerrouki
& Kouassi Kouassi

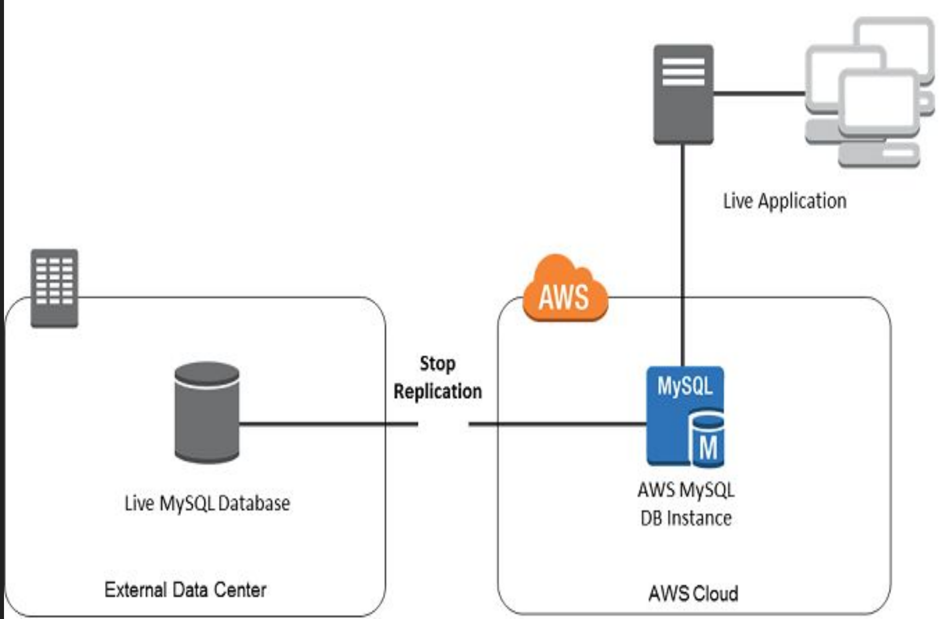
Database Application

- Our database was designed by gathering specific information of the Coronavirus in each state of the United States across five tables.
- The Database has one entity with the State as its primary key which all other entities use as both a foreign key and primary key when referencing that table making all but one entity a weak entity as the only defining relationship across each table is the State in which all of the information pertains to.
- This database is very simple and effective to implement because there are no multivalued dependencies and a limited number of functional dependencies. This makes the normalization process much easier to implement which makes designing the schema significantly easier. The main drawback is that it is required to dynamically extract information instead of manually entering data which will hopefully be rectified in the future.

ER Diagram


NumOfPeople
Density
CaseDensity
State
Population
DeathDensity
State
TotalCases
TotalDeaths
TotalTests
Contains
Nation
Correlates_To
State
TestsRatio
Ratio
DeathRatio
Controls
StartRate
EndRate
State
Infectionrate
Impacts
Affects
Workforce
State
Workers
Unemployment
Lockdown

Database Cloud Implementation



- The database is stored on the cloud using Amazon AWS and then connecting to the cloud which is implemented using MySql Workbench.

- The database uses the Free Tier with the smallest memory provided because this database doesn't require much memory in order to store information across 50 states.

- In order to connect to Amazon's cloud server using MySQL, we had to enable public accessibility of the server and also create a security group in which only our IP addresses provided by Amazon were included in that group.

- This way, only Nazim and Kouassi were authorized users to the database. This needed to be done including obtaining the key encrypted using SSH protocol in order to connect the database with MySQL.

Database FrontEnd/BackEnd

- The frontend was implemented using HTML and CSS whereas the backend was implemented using PHP which enables us to connect the frontend to our database in the Cloud using the MySQLI API.
- Using php, we were able to implement queries from our database and display specific pieces of data which can then be connected to the frontend whenever a user interacts with the database.
- Raw data was collected across numerous sites which will be included in the references. Multiple sources were used in order to establish consistency in the data that was used and because the database we were implementing required specific pieces of information that wasn’t available from a single source. This was done so that people can draw connections and correlate specific data to other variables such as the impact population and population density has on the current infection rate of a state.

Database Schema

Views: Views were used in order to capture specific, generalized information that users would want to see. This includes identifying which state has the maximum and minimum total cases as well as displaying a collection of states whose total number of cases were above or below the average number of total cases across all states.

Procedures: Procedures were used whenever the user wants to input certain values. These values are used as parameters so we can precompile queries and use these parameters in order to display data the user wants to see. This was often used in order to identify relationships between the population and rate of infections in addition to the number of cases accrued or the amount of deaths and tests that states were doing in proportion to the number of the cases. These procedures used threshold values as inputting specific numbers doesn't guarantee any data being outputted otherwise.

Triggers: Triggers were used because there was a functional dependencies across two tables. As such, insertions and modifications made on one table can be made on another table by using the values in the first table in order to compute the values in the second table. This was done across the Nation and Ratio table.

Indexes: Indexes were only used on the primary key for basic searches. Otherwise, we considered using indexes on the total number of cases, tests, and population size as a lot of searches were based on this. However, we decided against it because indexes cause overhead whenever modifications are being made and for such a relatively small database, the overhead is not worth it. Searches will be fast regardless because of that.

Privileges: We only had one admin user who was granted all privileges and had access to triggers once binary logging was enabled.

Functionality(Admin)

1. Users with a corresponding username and password are entered within a database to facilitate proper login authentication.
2. Authorized users will be able to login and reset their password if needed at their convenience without an admin having to manually reset the password. This is crucial when multiple users are working on the database concurrently.
3. Authorized users can add, delete, and modify any of the database within the database. Exceptions will be thrown when data or lack of data is provided in order to effectively add, delete, or modify data.

Functionality(Frontend)

1. Users will be able to conduct basic searches. These include searching for total cases, deaths, and tests across all states or a single state as well as searching for basic population, infection rate, and state of employment for each state.
2. Views will provide basic information such displaying states with the maximum and minimum cases as well as states that are above and below the number of average cases. It will also provide a relationship between the ratio between tests and cases conducted and the unemployment percentage for states whose daily infection rates are decreasing.
3. Users will be able to input certain thresholds for total cases, populations, death ratios, and number of tests and test ratios to identify key relationships that are crucial in understanding the spread of the virus.

Conclusion

This was an enjoyable project to make and a significant learning experience for the both of us. Learning to create queries is very important, but understanding the nuances behind full stack development was even more important for us considering we had no experience in such a field at all and had to do hefty research before diving in. As such, now we understand the intricacies behind full stack development and understand what information to search for when learning and understanding different frameworks.

References

https://usafacts.org/visualizations/coronavirus-covid-19-spread-map/

https://www.uschamber.com/series/above-the-fold/unemployment-claims-amid-coronavirus-state-state-analysis

https://www.nytimes.com/interactive/2020/us/coronavirus-us-cases.html

https://www.vox.com/2020/3/26/21193848/coronavirus-us-cases-deaths-tests-by-state

https://www.worldometers.info/coronavirus/country/us/

https://worldpopulationreview.com/states/state-densities/