**Final Project Proposal Report**

**Course: Database Design**

**Instructor: Donald Strawser**

**Team Members**

1. Himani Borana : hborana@iu.edu
2. Diksha Singh : singhdi@iu.edu
3. Sumukha Gargesh Pulipak : supuli@iu.edu
4. Manideep Regonda : mregonda@iu.edu

**Project Title: Covid-19 Vaccination Analysis**

**Overview**

In our quest to understand and respond effectively to the challenges posed by COVID-19, our motivation is deeply rooted in the lessons learned from historical events, particularly the Spanish Flu that occurred nearly a century ago. Back then, the lack of vaccinations and comprehensive analysis left us vulnerable during a global health crisis. The hindsight of this historical gap propels us to conduct a thorough analysis of COVID-19 vaccinations today.

Our primary objective is to derive insights that not only aid our response to the current pandemic but also lay the groundwork for future pandemics. By delving into the intricacies of vaccination distribution, coverage, and conducting comparative evaluations in states like California and Washington, we aspire to contribute to a more proactive and informed approach to public health strategies. This analysis is driven by a forward-looking perspective, aiming to build resilience and preparedness for potential health challenges in the years to come.

**Community - (Washington & California)**

* The focus of our project is the population in the United States, specifically in the states of Washington and California, who are receiving COVID-19 vaccinations. We are analyzing vaccination data to gain insights into the vaccination progress, distribution, rates, and trends within these regions.
* We chose Washington and California as the focus of our project as California is the most populated state in the USA and chose Washington to perform analysis because we needed a state which is neither highest populated nor least populated.
* By focusing on regions with such data, we can ensure the quality and depth of our research, making it easier to derive insights and make informed decisions based on the available information. This approach allows us to work with the data we have effectively and efficiently.

**Population -**

* **California -** 38,940,231 (~39 million approx.)( As of 2023)
* **Washington -** 7,951,150 (~7.9 million approx.) (As of 2023)

**Dataset - United States Covid-19 Vaccinations Data**

**Link:**<https://www.kaggle.com/datasets/sandhyakrishnan02/united-states-covid19-vaccinations/>

In our study of COVID-19 vaccinations, we used a "United States COVID-19 Vaccination" initially with 48,000 records and 17 attributes, which covers approximately a two-year span (From 13th January 2021 to 1st February 2023). To analyze key aspects like distribution and coverage, we narrowed it down to 12 columns and 1500 rows. We specifically focused on California and Washington because these states had complete and detailed data compared to others. This deliberate choice ensures a thorough examination and valuable insights. While we concentrate on these states, there's potential to expand our analysis to other areas later, maintaining a strategic approach for robust public health strategy enhancement.

**Location**: This column specifies the geographic area for which vaccination data is recorded, including states and federal entities.

**Date**: The date of observation or reporting, allowing for temporal analysis of vaccination trends.

**Total Vaccinations**: This column keeps track of the total number of COVID-19 vaccine doses administered. It counts each dose individually, making it a valuable metric for assessing the volume of vaccine administration.

**Total Vaccinations per Hundred**: This metric calculates the total vaccinations per 100 people in the population, providing a standardized measure for comparing vaccination rates across regions.

**Daily Vaccinations**: This metric provides the daily average of new doses administered, smoothed over a 7-day period. It's a more stable measure for assessing daily vaccination rates.

**Daily Vaccinations per Million**: This metric standardizes daily vaccinations by calculating them per 1,000,000 people in the population, facilitating comparisons across regions of different sizes.

**People Vaccinated**: This column counts the total number of individuals who have received only one vaccine dose. It's a valuable metric for monitoring the proportion of the population that has started the vaccination process and did not take the second dose.

**People Vaccinated per Hundred**: This metric expresses the number of people vaccinated per 100 people in the population, offering a percentage measure of the proportion that has received at least one dose.

**People Fully Vaccinated**: This metric tracks the total number of individuals who have completed the full vaccination regimen prescribed by the initial vaccination protocol.

**People Fully Vaccinated per Hundred**: Similar to the previous metric, this one calculates the percentage of the population that is fully vaccinated, representing a crucial measure of population protection.

**Total Boosters**: This column tracks the total number of COVID-19 vaccination booster doses administered, including those doses administered beyond the number prescribed by the initial vaccination protocol.

**Total Boosters per Hundred**: Similar to the total boosters metric, this one calculates the number of booster doses administered per 100 people in the population.

**Normalized Tables**

After normalization of our dataset, we identified the below 6 tables -

**1. States -** Contains attributes StateId and StateName.

**2. PeoplePartiallyVaccinated -** ID, Date, PartiallyVaccinated, StateID.

**3. PeopleFullyVaccinated -** Contains records of individuals who are partially vaccinated, i.e., have received only one vaccine dose. Contains attributesId, FullyVaccinatedPerHundred, FullyVaccinated, Date, StateId

**4. VaccinationRecords -** Contains record of total number of doses administered on a particular day. **Contains attributes**: RecordId, TotalVaccinations, Date, StateId

**5. VaccinationDailyRecord -** Contains records of daily vaccination data. Contains attributes - Date, DailyChangeInNoOfDoses, DailyVaccinationPerMillion, DailyVaccinations, StateId

**6. People\_BoosterShots -** Contains records of total no. of booster shots administered each day. Contains attributes - Id, FullyVaccinatedPerHundred, FullyVaccinated, Date, StateId

[For more detailed information, refer to the ER diagram and Relational Schema.]

**Problem Statements**

1. **Total Vaccination Calculation:** Find the overall count of COVID-19 vaccine doses administered in California and Washington states.
2. **Daily Vaccination Trends:** Determine the average daily vaccination rate over a specific timeframe and present the trends visually.
3. **Latest Vaccination Status:** Retrieve the most recent data from the database to identify the current numbers of fully vaccinated and partially vaccinated individuals in each state.
4. **Population Percentage Analysis:** Calculate the percentages of the population that are fully vaccinated and partially vaccinated in California and Washington states.
5. **Booster Dose Percentage**: Determine the percentage of the population in California and Washington that has received a COVID-19 vaccine booster dose.

These along with other questions were formulated subsequent to a comprehensive review and deep understanding of the dataset. Our approach involved a thorough examination of the data's implications. We aimed to identify areas where the available data could offer solutions to specific problems or provide valuable insights. These questions were designed to address key issues and serve as a foundation for potential improvements in the vaccination efforts in Washington and California. Our objective was to leverage the data to derive meaningful insights that could inform and enhance public health strategies in these regions.

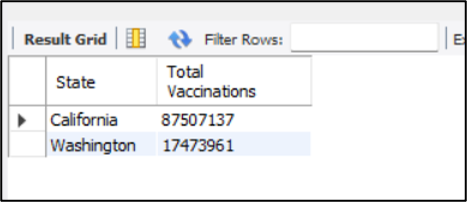
**Explanation of Solutions/Queries for the above problem statements**

**1- Calculate the total COVID-19 doses administered in California and Washington.**

| SELECT   st.StateName as State, vcr.TotalVaccinations as 'Total Vaccinations' FROM   State st JOIN   VaccinationRecord vcr ON st.StateId = vcr.StateId JOIN   (SELECT   StateId, MAX(Date) AS RecentDate  FROM   VaccinationRecord  GROUP BY   StateId) AS latest\_dt\_vr ON vcr.StateId = latest\_dt\_vr.StateId AND vcr.Date = latest\_dt\_vr.RecentDate; |
| --- |

**Solution :**

Using the ‘State’ and ‘VaccinationRecord’ table, this SQL query obtains the total vaccination doses administered in Washington and California states. The accuracy of the results can be confirmed by verifying the results against online resources.

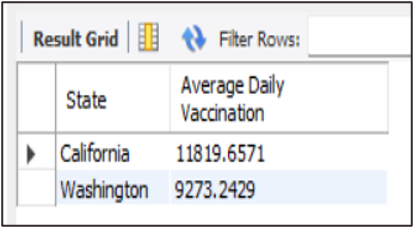
****

**2- What is the average daily vaccination over a specific time period?**

| SELECT  st.StateName as State,  AVG(vdr.DailyVaccinations) AS 'Average Daily Vaccination' FROM  VaccinationsDailyRecord vdr JOIN State as st ON vdr.StateId = st.StateId WHERE  vdr.Date BETWEEN '2022-03-18' AND '2022-05-26' GROUP BY  st.StateId, st.StateName; |
| --- |

**Solution:**

The average daily vaccination rate for each state within the given time frame (‘2022-03-18’ to ‘2022-05-26’) (2 months) is determined by the given SQL query. It groups the results by the state via joining the “State” and “VaccinationDailyRecord” tables. The result displays state name and the corresponding average daily vaccination count.

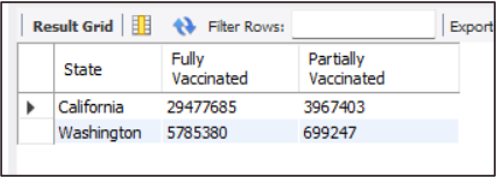


**3- What are the latest recorded numbers of fully vaccinated and partially vaccinated individuals in each state?**

| **SELECT  st.StateName as State, pfvc.RecentFullyVaccinated as 'Fully Vaccinated', ppvc.RecentPartiallyVaccinated as 'Partially Vaccinated' FROM State st LEFT JOIN  (SELECT  StateId, FullyVaccinated AS RecentFullyVaccinated, Date AS FullyVaccinatedDate  FROM PeopleFullyVaccinated pfvc1  WHERE  pfvc1.Date = (SELECT MAX(pfvc2.Date) FROM PeopleFullyVaccinated pfvc2 WHERE pfvc2.StateId = pfvc1.StateId)  ) AS pfvc ON st.StateId = pfvc.StateId LEFT JOIN  (SELECT  StateId, PartiallyVaccinated AS RecentPartiallyVaccinated, Date AS PartiallyVaccinatedDate  FROM  PeoplePartiallyVaccinated ppvc1  WHERE  ppvc1.Date = (SELECT MAX(ppvc2.Date) FROM PeoplePartiallyVaccinated ppvc2 WHERE ppvc2.StateId = ppvc1.StateId)  ) AS ppvc ON st.StateId = ppvc.StateId ORDER BY st.StateName;** |
| --- |

**Solution:**

The query retrieves the most recent data on the number of people in each state who are fully and partially vaccinated.The “PeopleFullyVaccinated” and “PeoplePartiallyVaccinated” tables are queried to extract the most recent data, which is then sorted by the state names using left joins with subqueries.

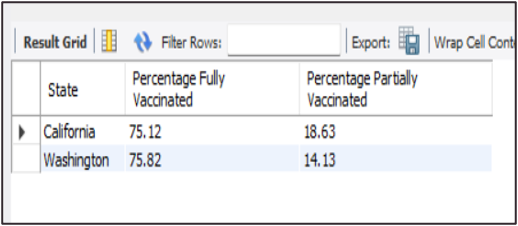


**4- What is the percentage of the population that are fully vaccinated and partially vaccinated in the states of California and Washington ?**

| **SELECT st.StateName as State,  ROUND( IFNULL( (fvc.RecentFullyVaccinated /  CASE  WHEN st.StateName = 'California' THEN 39240000  WHEN st.StateName = 'Washington' THEN 7630000  END) \* 100, 0), 2) AS 'Percentage Fully Vaccinated',  ROUND( IFNULL(  (pvc.RecentPartiallyVaccinated /  CASE  WHEN st.StateName = 'California' THEN 39240000  WHEN st.StateName = 'Washington' THEN 7630000  END) \* 100, 0), 2) AS 'Percentage Partially Vaccinated' FROM State st LEFT JOIN (  SELECT StateId, MAX(FullyVaccinated) AS RecentFullyVaccinated FROM PeopleFullyVaccinated  GROUP BY StateId) AS fvc ON st.StateId = fvc.StateId LEFT JOIN (  SELECT StateId, MAX(PartiallyVaccinated) AS RecentPartiallyVaccinated  FROM PeoplePartiallyVaccinated  GROUP BY StateId ) AS pvc ON st.StateId = pvc.StateId;** |
| --- |

**Solution:**

This SQL query determines the percentage of people in the states of California and Washington who are fully and partially vaccinated. The most recent vaccination data is obtained using subqueries ,and the percentages are computed based on the population of each state.

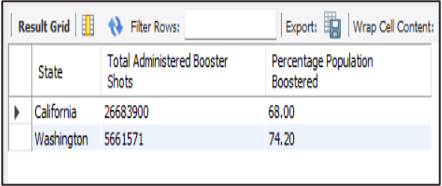


**5- What is the percentage of the population in California and Washington that has received a COVID-19 vaccine booster dose ?**

| SELECT  st.StateName as State, BoosterShotsData.TotalBoosterShotsAdministered as 'Total Administered Booster Shots',   ROUND((  BoosterShotsData.TotalBoosterShotsAdministered /   CASE   WHEN st.StateName ='California' THEN 39240000  WHEN st.StateName ='Washington' THEN 7630000  ELSE 1  END  ) \* 100, 2) AS 'Percentage Population Boostered' FROM State st JOIN (  SELECT StateId, TotalBoosterShots AS TotalBoosterShotsAdministered, Date  FROM People\_BoosterShots  WHERE (StateId, Date) IN (  SELECT StateId, MAX(Date) FROM People\_BoosterShots GROUP BY StateId  ) ) AS BoosterShotsData ON st.StateId = BoosterShotsData.StateId; |
| --- |

**Solution:**

The above SQL query determines the percentage of people in California and Washington who have gotten a booster dose of the COVID-19 vaccine. The query obtains the most current booster shot data from the ‘People\_BoosterShots’ table and computes the percentage based on the populations of the corresponding states.

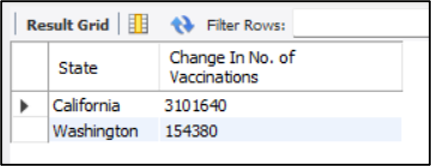


**6- How has the total number of vaccinations changed from one date to another in both states ?**

| **SELECT  st.StateName as State,  (EndDateVaccinations.TotalVaccinations - StartDateVaccinations.TotalVaccinations) AS 'Change In No. of Vaccinations' FROM State st JOIN  (SELECT StateId, TotalVaccinations FROM VaccinationRecord WHERE Date ='2021-09-16') AS EndDateVaccinations  ON st.StateId = EndDateVaccinations.StateId JOIN  (SELECT StateId, TotalVaccinations FROM VaccinationRecord WHERE Date ='2021-08-13') AS StartDateVaccinations  ON st.StateId = StartDateVaccinations.StateId;** |
| --- |

**Solution:**

This SQL query determines how many immunizations were administered overall in California and Washington between two given dates. The “VaccinationRecord” table is joined twice: once for the start date(‘2021-08-13’) and once for the finish date(‘2021-09-16’), and then the query computes the difference in the total injections.

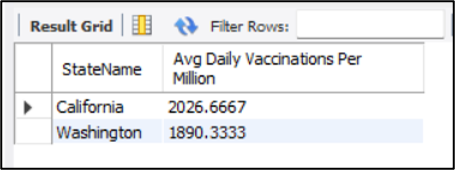


**7- What is the daily average number of vaccinations administered per million over the given time period in California and Washington ?**

| SELECT   st.StateName,  AVG(vdr.DailyVaccinationPerMillion) AS 'Avg Daily Vaccinations Per Million' FROM VaccinationsDailyRecord vdr JOIN   State st ON vdr.StateId = st.StateId WHERE   vdr.Date BETWEEN '2021-01-13' AND '2021-01-15' GROUP BY   st.StateName; |
| --- |

**Solution:**

The query determines , average daily vaccination rate per million in California and Washington for the designated time frame (‘2021-01-13’ to ‘2021-01-15’) . In order to get the average for each state, it connects the ‘VaccinationsDailyRecord’ and ‘State’ database and groups the results by the State names .

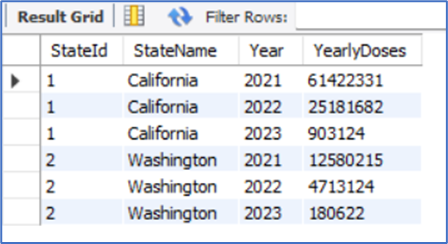


**8. How many vaccination doses were administered each year for each state?**

| SELECT vcr1.StateId, st.StateName,  YEAR(vcr1.Date) as Year, (vcr1.TotalVaccinations - COALESCE(vcr2.TotalVaccinations, 0)) as YearlyDoses FROM VaccinationRecord vcr1 LEFT JOIN  VaccinationRecord vcr2 ON vcr1.StateId = vcr2.StateId AND vcr2.Date = (  SELECT MAX(vcr3.Date)  FROM VaccinationRecord vcr3  WHERE YEAR(vcr3.Date) = YEAR(vcr1.Date) - 1 AND vcr3.StateId = vcr1.StateId  ) JOIN State st ON vcr1.StateId = st.StateId WHERE  vcr1.Date IN (SELECT MAX(vcr4.Date) FROM VaccinationRecord vcr4 GROUP BY YEAR(vcr4.Date), vcr4.StateId) ORDER BY vcr1.StateId, Year; |
| --- |

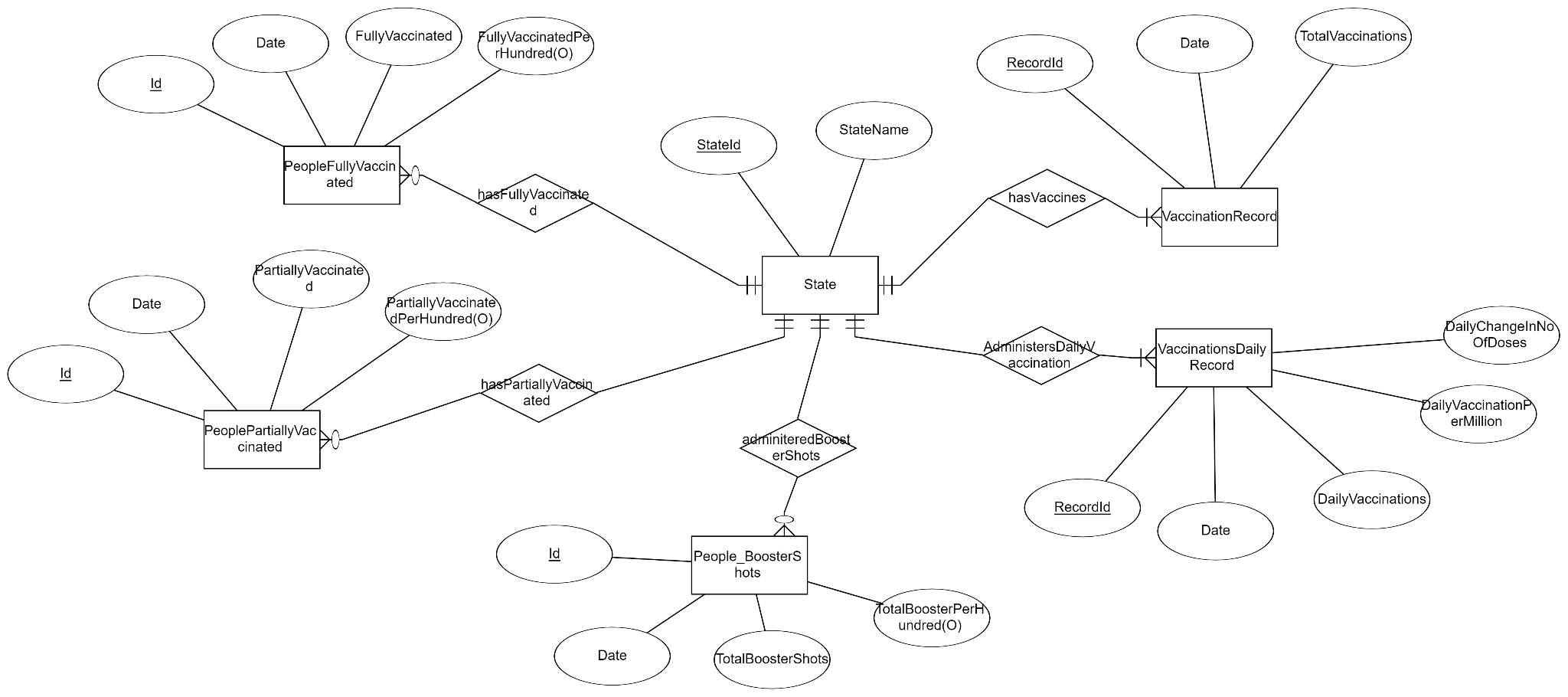
**Solution:**

By comparing the total vaccinations received in a particular year with the total vaccinations received in the previous year, this SQL query determines the annual vaccination doses administered for each state. The pertinent information is retrieved via joins and subqueries, and the StateId, StateName , Year, and associated annual dosages are displayed.

****

**ER and Relational Diagrams**

**ER Diagram:**

****

In the above ER diagram, we have the following entities:

**1. States -** Contains the state id and state names.

**2. PeoplePartiallyVaccinated -** Contains records of individuals who are partially vaccinated, i.e., have received only one vaccine dose.

**3. PeopleFullyVaccinated -** Contains records of individuals who are fully vaccinated i.e., have received both vaccine doses.

**4. VaccinationRecords -** Contains record of total number of doses administered on a particular day. It may not be equal to the total number of people vaccinated. For example - if an individual receives one vaccine dose, the count goes up by one, and on receiving the second dose, it goes up by 1 again.

**5. VaccinationDailyRecord -** Contains records of daily vaccination data.

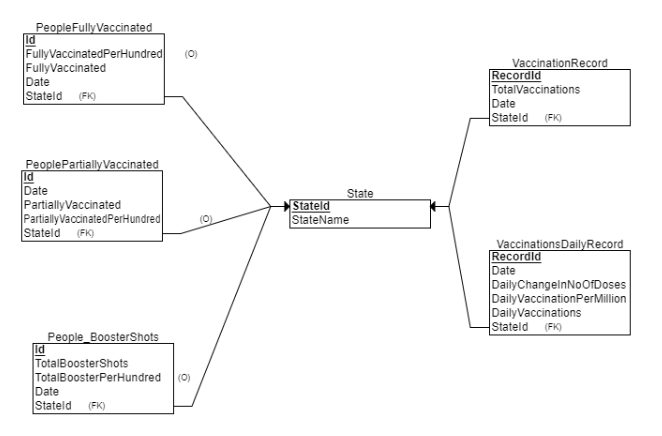
**6. People\_BoosterShots -** Contains records of total no. of booster shots administered each day.

Below we’ve explained the different relationships between different entities:

* **‘hasVaccines’** connects ‘VaccinationRecord’ to ‘State’, indicating that a state has one or more vaccination records.
* **‘hasFullyVaccinated’** and **‘hasPartiallyVaccinated’** connect the ‘PeopleFullyVaccinated’ and ‘PeoplePartiallyVaccinated’ to the ‘State’ entity, suggesting that each state has records of fully and partially vaccinated people.
* **‘administeredBoosterShots’** connects ‘People\_BoosterShots’ to ‘State’, indicating that a state has records of people who received booster shots.

**Relational Schema:**

The below Relational Schema is derived from the above ER diagram. This database would allow one to query information about vaccination rates, both fully and partially, as well as booster shots, a distribution, coverage and comparisons of vaccinations for Washington and California states.

****

**Ethical and Privacy Considerations**

**​​**The “United States COVID-19 Vaccination” dataset is open-source and is city specific and doesn’t have any individual information. Nevertheless, we are committed to the following ethical measures:

* **Data Anonymization:** By making private information anonymous, we will lessen the possibility of identifying a specific person.
* **Data Integrity:** We will preserve the accuracy and reliability of the data to ensure its meaningfulness for analysis.
* **Responsible User:** We won’t use data in any way that is damaging or discriminatory; instead, we’ll use it to enhance public health while abiding by ethical standards.
* **Transparency:** We’ll keep lines of communication open and transparent regarding how we use data and what our research goals are.
* **Open-Source Principles:** We shall respect the dataset’s open-source status by contributing positively to the community and sharing insights.

**Challenges Faced**

1. We implemented a process to import data from CSV files seamlessly into our project. Additionally, we developed a script that automates the creation of tables based on a thorough analysis of the imported data.
2. To address null values within the dataset, we devised strategies that align with the context of the data.
3. For specific columns such as total\_vaccinations, FullyVaccinatedPerHundred, FullyVaccinated, TotalBoosterShots, TotalBoosterPerHundred, prev\_people\_vaccinated, and prev\_people\_fully\_vaccinated, we utilized the previous appropriate values when the corresponding data was missing.
4. In cases where Daily vaccination data was absent, we assumed no vaccinations occurred on that day and assigned 0 to daily\_vaccinations\_raw, daily\_vaccinations\_per\_million, and daily\_vaccinations.
5. We prioritized the accuracy and reliability of the dataset by meticulously processing the data as outlined above. To maintain real-time accuracy, we developed a script that facilitates the population of the database with the most recent CSV updates.

**Scripts Used for Dataset Preprocessing**

Below are the scripts to create tables and import data from CSV.

**Create\_Tables.py**

| import mysql.connector  # Connect to MySQL server mydb = mysql.connector.connect(  host="localhost",  user="root",  password="sumukha96", )  # Create a cursor mycursor = mydb.cursor()  # Create the database if it doesn't exist mycursor.execute("CREATE DATABASE IF NOT EXISTS Final\_DB") print("CREATE DATABASE IF NOT EXISTS Final\_DB;")  # Switch to the database mycursor.execute("USE Final\_DB") print("USE Final\_DB;")  # Create tables mycursor.execute("""  CREATE TABLE State (  StateId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  StateName VARCHAR(255) NOT NULL  ) """) print("CREATE TABLE State (StateId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,StateName VARCHAR(255) NOT NULL);") # Create tables mycursor.execute("""  CREATE TABLE VaccinationRecord (  RecordId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  TotalVaccinations INT NOT NULL,  Date VARCHAR(255) NOT NULL,  StateId INT NOT NULL,  FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT  ) """) print("CREATE TABLE VaccinationRecord (RecordId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,TotalVaccinations INT NOT NULL,Date VARCHAR(255) NOT NULL,StateId INT NOT NULL,FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT);")  mycursor.execute("""  CREATE TABLE PeopleFullyVaccinated (  Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  FullyVaccinatedPerHundred INT,  FullyVaccinated INT NOT NULL,  Date VARCHAR(255) NOT NULL,  StateId INT NOT NULL,  FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT  ) """) print("CREATE TABLE PeopleFullyVaccinated (Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, FullyVaccinatedPerHundred INT, FullyVaccinated INT NOT NULL, Date VARCHAR(255) NOT NULL, StateId INT NOT NULL, FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT);")  mycursor.execute("""  CREATE TABLE PeoplePartiallyVaccinated (  Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  Date VARCHAR(255) NOT NULL,  PartiallyVaccinated INT NOT NULL,  StateId INT NOT NULL,  FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT  ) """)  print("CREATE TABLE PeoplePartiallyVaccinated (Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, Date VARCHAR(255) NOT NULL, PartiallyVaccinated INT NOT NULL, StateId INT NOT NULL, FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT);")   mycursor.execute("""  CREATE TABLE People\_BoosterShots (  TotalBoosterShots INT NOT NULL,  TotalBoosterPerHundred INT,  Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  Date VARCHAR(255) NOT NULL,  StateId INT NOT NULL,  FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT  ) """) print("CREATE TABLE People\_BoosterShots (TotalBoosterShots INT NOT NULL, TotalBoosterPerHundred INT, Id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, Date VARCHAR(255) NOT NULL, StateId INT NOT NULL, FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT);")  mycursor.execute("""  CREATE TABLE VaccinationsDailyRecord (  RecordId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,  Date VARCHAR(255) NOT NULL,   DailyChangeInNoOfDoses INT NOT NULL,  DailyVaccinationPerMillion INT NOT NULL,  DailyVaccinations INT NOT NULL,  StateId INT NOT NULL,  FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT  ) """) print("CREATE TABLE VaccinationsDailyRecord (RecordId INT NOT NULL AUTO\_INCREMENT PRIMARY KEY, Date VARCHAR(255) NOT NULL, DailyChangeInNoOfDoses INT NOT NULL, DailyVaccinationPerMillion INT NOT NULL, DailyVaccinations INT NOT NULL, StateId INT NOT NULL, FOREIGN KEY (StateId) REFERENCES State(StateId) ON DELETE RESTRICT);")   # Insert data into State table mycursor.execute("INSERT INTO State (StateId, StateName) VALUES (1, 'California'), (2, 'Washington')") print("INSERT INTO State (StateId, StateName) VALUES (1, 'California'), (2, 'Washington');")  # Commit the changes mydb.commit()  # Close cursor and connection mycursor.close() mydb.close() |
| --- |

**2. import\_CSV.py**

| import csv import mysql.connector  # Connect to MySQL server mydb = mysql.connector.connect(  host="localhost",  user="root",  password="sumukha96" )  # Create a cursor mycursor = mydb.cursor() mycursor.execute("USE Final\_DB")  prev\_total\_vaccinations = 0 prev\_FullyVaccinatedPerHundred, prev\_FullyVaccinated = 0,0 prev\_TotalBoosterShots, prev\_TotalBoosterPerHundred = 0,0 prev\_people\_vaccinated ,prev\_people\_fully\_vaccinated = 0,0  # Read the CSV file with open('/Users/sumukha/Downloads/us\_state\_vaccinations.csv', 'r') as file:  reader = csv.reader(file)   # Skip the header row  next(reader, None)   for row in reader:   # Extract data from the row  state\_id = 0  date, location, total\_vaccinations, total\_distributed, people\_vaccinated, \  people\_fully\_vaccinated\_per\_hundred, total\_vaccinations\_per\_hundred, \  people\_fully\_vaccinated, people\_vaccinated\_per\_hundred, distributed\_per\_hundred, \  daily\_vaccinations\_raw, daily\_vaccinations, daily\_vaccinations\_per\_million, \  share\_doses\_used, total\_boosters, total\_boosters\_per\_hundred = row        if location == "California":  state\_id = 1  elif location == "Washington":  state\_id = 2  else:  continue       # Convert string values to float for calculation  try:  total\_vaccinations = float(total\_vaccinations)  except ValueError:  total\_vaccinations = None   try:  people\_fully\_vaccinated = float(people\_fully\_vaccinated)  except ValueError:  people\_fully\_vaccinated = None   try:  people\_vaccinated = float(people\_vaccinated)  except ValueError:  people\_vaccinated = None    if total\_vaccinations == None or total\_vaccinations == '':  total\_vaccinations = prev\_total\_vaccinations  else:  prev\_total\_vaccinations = total\_vaccinations   # Insert data into VaccinationRecord table  mycursor.execute("""  INSERT INTO VaccinationRecord   (Date, TotalVaccinations, StateId)   VALUES (%s, %s, %s)  """, (date, total\_vaccinations, state\_id))  mydb.commit() # Commit the changes    final\_query = f"INSERT INTO VaccinationRecord (Date, TotalVaccinations, StateId) VALUES ('%s', %s, %s);"  data = (date, total\_vaccinations, state\_id)  formatted\_query = final\_query % data  print(formatted\_query)    if people\_fully\_vaccinated\_per\_hundred == None or people\_fully\_vaccinated\_per\_hundred == '' or people\_fully\_vaccinated == None or people\_fully\_vaccinated == '':  people\_fully\_vaccinated\_per\_hundred, people\_fully\_vaccinated = prev\_FullyVaccinatedPerHundred, prev\_FullyVaccinated     else:  prev\_FullyVaccinatedPerHundred, prev\_FullyVaccinated = people\_fully\_vaccinated\_per\_hundred, people\_fully\_vaccinated    # Insert data into PeopleFullyVaccinated table  mycursor.execute("""  INSERT INTO PeopleFullyVaccinated   (Date, FullyVaccinatedPerHundred, FullyVaccinated, StateId)   VALUES (%s, %s, %s, %s)  """, (date, people\_fully\_vaccinated\_per\_hundred, people\_fully\_vaccinated, state\_id))  mydb.commit()  final\_query = f"INSERT INTO PeopleFullyVaccinated (Date, FullyVaccinatedPerHundred, FullyVaccinated, StateId) VALUES ('%s', %s, %s, %s);"  data = (date, people\_fully\_vaccinated\_per\_hundred, people\_fully\_vaccinated, state\_id)  formatted\_query = final\_query % data  print(formatted\_query)    if people\_vaccinated == None or people\_vaccinated == '' or people\_fully\_vaccinated == None or people\_fully\_vaccinated =='':  people\_vaccinated,people\_fully\_vaccinated = prev\_people\_vaccinated ,prev\_people\_fully\_vaccinated    else:  prev\_people\_vaccinated ,prev\_people\_fully\_vaccinated = people\_vaccinated,people\_fully\_vaccinated      # Insert data into PeoplePartiallyVaccinated table  mycursor.execute("""  INSERT INTO PeoplePartiallyVaccinated   (Date, PartiallyVaccinated, StateId)   VALUES (%s, %s, %s)  """, (date, people\_vaccinated - people\_fully\_vaccinated, state\_id))  mydb.commit() # Commit the changes  final\_query = f"INSERT INTO PeoplePartiallyVaccinated (Date, PartiallyVaccinated, StateId) VALUES ('%s', %s, %s);"  data = (date, people\_vaccinated - people\_fully\_vaccinated, state\_id)  formatted\_query = final\_query % data  print(formatted\_query)    if total\_boosters == None or total\_boosters == '' or total\_boosters\_per\_hundred == None or total\_boosters\_per\_hundred == '':  total\_boosters , total\_boosters\_per\_hundred = prev\_TotalBoosterShots, prev\_TotalBoosterPerHundred    else:  prev\_TotalBoosterShots, prev\_TotalBoosterPerHundred = total\_boosters , total\_boosters\_per\_hundred    mycursor.execute("""  INSERT INTO People\_BoosterShots   (Date, TotalBoosterShots, TotalBoosterPerHundred, StateId)   VALUES (%s, %s, %s, %s)  """, (date, total\_boosters, total\_boosters\_per\_hundred, state\_id))  mydb.commit()  final\_query = f"INSERT INTO People\_BoosterShots (Date, TotalBoosterShots, TotalBoosterPerHundred, StateId) VALUES ('%s', %s, %s, %s);"  data = (date, total\_boosters, total\_boosters\_per\_hundred, state\_id)  formatted\_query = final\_query % data  print(formatted\_query)      if daily\_vaccinations\_raw == None or daily\_vaccinations\_raw =='' or daily\_vaccinations\_per\_million == None or daily\_vaccinations\_per\_million == '' or daily\_vaccinations == None or daily\_vaccinations == '':  daily\_vaccinations\_raw, daily\_vaccinations\_per\_million, daily\_vaccinations = 0,0,0    # Insert data into VaccinationsDailyRecord table  mycursor.execute("""  INSERT INTO VaccinationsDailyRecord   (Date, DailyChangeInNoOfDoses, DailyVaccinationPerMillion, DailyVaccinations, StateId)   VALUES (%s, %s, %s, %s, %s)  """, (date, daily\_vaccinations\_raw, daily\_vaccinations\_per\_million, daily\_vaccinations, state\_id))  mydb.commit() # Commit the changes  final\_query = f"INSERT INTO VaccinationsDailyRecord (Date, DailyChangeInNoOfDoses, DailyVaccinationPerMillion, DailyVaccinations, StateId) VALUES ('%s', %s, %s, %s, %s);"  data = (date, daily\_vaccinations\_raw, daily\_vaccinations\_per\_million, daily\_vaccinations, state\_id)  formatted\_query = final\_query % data  print(formatted\_query)     mycursor.close() mydb.close() |
| --- |

**References**

1. Dataset:<https://www.kaggle.com/datasets/sandhyakrishnan02/united-states-covid19-vaccinations?resource=download>
2. Citation: <https://doi.org/10.1038/s41562-021-01122-8>
3. <https://www.rkimball.com/creating-a-database-design-for-the-software-development-lifecycle/>
4. <https://www.databasestar.com/sample-data-sql/>
5. <https://covid19.ca.gov/vaccination-progress-data/>
6. https://usafacts.org/visualizations/covid-vaccine-tracker-states/state/washington/
7. <https://www.mysql.com/products/workbench/>