COL362 Project

COVID-19 Dashboard using RDBMS

Sarthak Agrawal	Piyush Gupta	Navneel Singhal
2018CS10383	2018CS10365	2018CS10360

April 2021

1 Introduction

1.1 Motivation

The ongoing COVID pandemic is of utmost importance in the public health space, and it is more important to be informed about such an issue in the interest of public health. It is hard to find a reliable dashboard or source of information that systematically provides information at district granularity, as well as real-time data in that spirit. We hence tried to tackle this problem by creating such a dashboard ourselves.

1.2 Project Description

In this project, we created an RDBMS-based dashboard for COVID-19 related information specific to India. From the data source we already have multiple useful tables such as - states, districts, state districts, case time series, tested numbers, icmr labs, vaccine doses administered, vaccine data which we used in our database, and also included other govt. census datasets (such as to determine population of a particular state or district). For deciding on queries we took inspiration from already existing COVID dashboards and queries that they offer, as well as think of some new queries that they don't offer. Finally we have provided an "admin" panel where it is possible to interactively add more info or update existing info to the database using SQL insert/update/delete statements and triggers.

1.3 Schema Diagram

2 Data

2.1 Data Sources

We have collected the data from the following sites

1. https://api.covid19india.org/
   (a) cowin_vaccine_data_statewise.csv

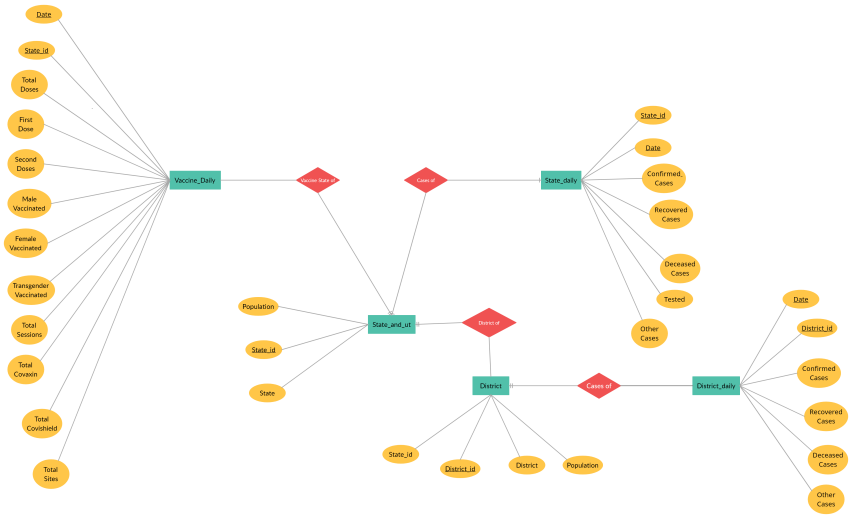


Figure 1: ER Diagram

(b) states.csv

(c) districts.csv

(d) statewise_tested_number_data.csv

(e) district_wise.csv

2. https://www.kaggle.com/danofer/india-census?select=india-districts-census-2011.csv

   (a) india_district_census.csv

2.2 Data Collection Methodology

The Data was readymade, just it was not perfect to be used directly so we have to do a cleanup before using it. The clean process is described in the next subsection.

2.3 Data Cleanup

We had done the data cleanup before using the data, this was required due to multiple reasons. The reasons are described below:

1. In districts.csv every row represent the number of cumulative cases of different city on a particular date. But the problem in this csv file was that there were many dates in which data was not given of every city, but of some cities.

2. In states.csv every row represent the number of cumulative cases of different city on a particular state. here the problem was same as of the above, on some dates the data is not given for all states.

3. The same problem as the above two was in the cowin_vaccine_data_statewise.csv file.

4. As the population was taken from different source so there were many districts whose name were spelled different and also the number of districts were different.

5. The most important problem was that the primary key of every table was the name of district or state, not any id. So it was consuming more data than required (as well as data redundancy because of unnormalized data).

6. Some tables were not perfectly normalized, so to reduce all redundancy we decided to keep each table in strictly bcnf form.

So to clean the data we have made a python file, and run it to clean the data. Now it handle the above issues and created 5 csv files ready to use. I will describe each file below

1. state_and_ut.csv: The columns in this csv file are state_id, state_name and the population of different state. There was no state_id in the real data so we created a unique id for each state.

2. district.csv : The columns in this csv file are district_id,state_id, district_name, and the population. Here also there was no district_id in the real data so we created a unique id for each district. For the populataion part we took the district described in the covid database as default district and tried to find the population from the cesnsus.csv files and if i could find the file then we set the population to that value or else null.

3. district_daily.csv: The columns in this csv files are district_id, date, and cases of different type. In the real data the values were given of cumulative and also on many dates the data was missing for some of the districts. So in this we assumed the data to be same as of the last previous date. and then subtracted the data to get the daily values from the cumulative values.

4. state_daily.csv: The columns in this csv files are state_id, date, and cases of different type. In the real data the values were given of cumulative and also on many dates the data was missing for some of the states. So in this we assumed the data to be same as of the last previous date. and then subtracted the data to get the daily values from the cumulative values.

5. vaccine_daily.cav: The columns in this csv files are state_id, date, and many columns of stats related to vaccine. In the real data the values were given of cumulative and also on many dates the data was missing for some of the states. So in this we assumed the data to be same as of the last previous date. and then subtracted the data to get the daily values from the cumulative values.

2.4 Data Statistics

As the table are created using merging some tables and creating the ids so we can't describe the size by csv. The total size of the data before cleanup was 15564 kb

3 Project Design

3.1 User View of the system

On the top of page the user sees a navigation bar, through which the user may select one of the 4 screens : Dashboard, State-wise, District-wise and Admin Panel. The description of each of the

Covid-19 India Dashboard

Dashboard | State-wise | District-wise | Admin Panel

Summary

Refresh

India covid-19 stats from 30 / 01 / 2020 to 25 / 03 / 2021

Confirmed Cases	Recovered Cases	Active Cases	Deceased Cases	Tested	Total Vaccine Doses
11787013	11229591	391956	160726	255481629	48354611

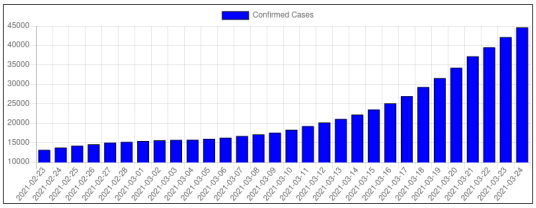
Columns Displayed :

☑ Confirmed Cases ☑ Recovered Cases ☑ Active Cases ☑ Deceased Cases ☐ Other Cases ☑ Tested ☑ Total Vaccine Doses ☐ Active Ratio
☐ Recovery Ratio ☐ Case Fatality Ratio ☐ Test Positivity Ratio ☐ Confirmed per lakh ☐ Recovered per lakh ☐ Active per lakh
☐ Deceased per lakh ☐ Other per lakh ☐ Tested per lakh ☐ Total Vaccine Doses per lakh ☐ Population

Daily Statistics

Refresh

7-day Moving Average | Confirmed Cases | for last 30 days



Vaccine Progress

Refresh

India vaccine progress from 16 / 01 / 2021 up to 25 / 03 / 2021

Total Sessions Conducted	Percentage Vaccinated (first dose)	Total Dose per lakh	First Dose per lakh	Second Dose per lakh
2074457	83.54	3526.03	2945.52	580.52

Columns Displayed :

☐ Total Dose ☐ First Dose ☐ Second Dose ☐ Males Vaccinated ☐ Females Vaccinated ☐ Transgender Vaccinated ☑ Total Sessions Conducted
☐ Total Covaxin ☐ Total Covishield ☑ Percentage Vaccinated (first dose) ☑ Total Dose per lakh ☑ First Dose per lakh ☑ Second Dose per lakh

States at a glance

Refresh

Click on any state to view more information about that state.

Sort states by Confirmed Cases | Descending

Name	Confirmed Cases	Active Cases	Recovered Cases	Deceased Cases	Tested	Vaccinated
Maharashtra	2564881	247299	2262593	53684	18725307	4159836
Kerala	1109909	24265	1080803	4528	12810707	2025879
Karnataka	975955	16886	946589	12461	20674133	2934392
Andhra Pradesh	895121	2946	884978	7197	14840401	1959874
Tamil Nadu	871440	9746	849064	12630	19011118	2307030
Delhi	651227	4890	635364	10973	14056463	850986
Uttar Pradesh	609443	4388	596286	8769	33972718	4852942
West Bengal	581865	3782	567771	10312	9015071	4141274
Odisha	339246	937	336337	1972	8861380	1886854
Chhattisgarh	329694	11934	313749	4011	5542288	1503824
Rajasthan	327175	4672	319695	2808	6726815	4789237
Telangana	304298	3352	299270	1676	9789113	914205
Gujarat	292169	8823	278880	4466	12926709	4259053
Haryana	282569	6745	272714	3110	6107247	1108229
Madhya Pradesh	280289	10047	266323	3919	6214522	2794468
Bihar	263940	726	261648	1565	23322388	2146198
Punjab	220276	20522	193280	6474	5703944	638926
Assam	218099	372	215277	1103	7155821	864607
Jammu and Kashmir	129031	1513	125535	1983	5830758	642804
Jharkhand	121695	969	119626	1100	5802842	1390474
Uttarakhand	98880	1112	94634	1706	2669701	561615
Himachal Pradesh	61301	1654	58615	1014	1216320	370585
Goa	56981	1156	55004	821	530780	100269
Puducherry	40645	586	39380	679	659537	54656
Tripura	33476	43	33021	389	635381	470340
Manipur	29362	53	28935	374	571447	103952
Chandigarh	25130	2178	22587	365	298087	61648
Arunachal Pradesh	16842	1	16785	56	412521	79400
Meghalaya	14019	22	13848	149	395112	101847
Nagaland	12226	2	11979	91	135549	72611
Ladakh	9942	101	9711	130	110068	45777
Sikkim	6210	44	5934	135	82162	69612
Andaman and Nicobar Islands	5041	6	4973	62	308812	18837
Mizoram	4452	16	4425	11	247878	66888
Dadra and Nagar Haveli and Daman and Diu	3485	63	3390	2	72410	0
Lakshadweep	699	105	588	1	46119	5482
State Unassigned	0	0	0	0	0	0

Data Analysis

Refresh

Find top 3 Days between 18 / 01 / 2021 and 25 / 03 / 2021 when Daily | Confirmed Cases were Maximum

Date	Value
2021-03-24	53419
2021-03-23	47239
2021-03-21	47009

Figure 2: Screenshot of dashboard page (Firefox on Ubuntu)

	Name	# of tuples	Time to load(ms)	Size after Cleanup(in kb)
Tables	state_and_ut	37	3.129	18
	district	801	45.925	0.9
	state_daily	12322	188.945	384
	district_daily	266733	4371.302	6607
	vaccine_daily	2516	49.230	153

Table 1: Data Statistics

screen (page) is as follows. Please also refer to the attached screen shot of dashboard page for better understanding.

- **India Dashboard (/dashboard)** This is the default page that user sees when he opens the webpage. This page deals with queries whose results are displayed in the granularity of entire India. The page is divided into multiple sections (5 in total), each section is dedicated to one "class" of queries. Here by a "class" we mean a collection of queries which are semantically similar but may differ with each other in implementation. Each section has some (or many) customizable parameters and attributes that the user may select to specialize their query. In this page the following sections are available
  - **Summary (/api/india/summary):** User selects a from and to date, and the database returns aggregate stats of India related to COVID-19 in that time interval. Here some stats such as number of confirmed cases, etc. are directly inferred from underlying tables, whereas some other stats such as active ratio or per lakh estimates are computed on the fly. As the number of stats are too large to display comfortably in one screen, user is provided with option to select precisely which columns he would like to retrieve.
  - **Daily Stats (/api/india/daily):** User selects a parameter that he is interested in (such as number of confirmed cases) and then a time window (such as last 7 days) and the database returns values of that parameter for each day in the time window. The result from the database is displayed in form of bar graph for convenience. The user can also select how the parameter should be computed. To this end, he has 3 options : Daily, Cumulative or 7-Day Moving Average. Internally, each option is mapped to a separate sql query.
  - **Vaccine Summary (/api/india/vaccine):** Similar to the first section of this page, this section also allows user to specify a time interval and the database returns aggregate stats in this time interval. The difference is that this section deals with stats related to vaccinations.
  - **States at a glance (/api/india/liststates):** In this section the user can view a list of all states of India alongwith their key stats. User can choose to sort the list in ascending or descending order with respect to a parameter of his choice. To get detailed stats about a state, the user should visit the State-wise screen from the navigation bar.
  - **Analysis (/api/india/analysis):** This section allows user to fire some miscellaneous queries of a particular form. The user can select a time interval and a parameter of his choice and request from database list of top 3 days when the parameter was either

maximum or minimum. As in the daily stats query, the user again has a choice of customizing how we would want the parameter to be computed : Daily, Cumulative or 7-day moving average.

- **State-wise (/states)** This page is very similar to the previous page and all the sections that appeared in the previous page also appear in this page. All stats displayed in this screen are with state-level granularity. This screen has following sections of intereset
  - **Select State (/api/states/list):** First the user must select a state from the available list of states from dropdown. Once selected the user presses GO button and the rest of page is then rendered.
  - **Others (/api/states/summary, /api/states/daily, /api/states/vaccine/, /api/states/listdistricts, /api/states/analysis):** Now there are 5 more sections which are similar to the sections in the previous screen (india dashboard) the only difference is of course that this time the stats are with respect to the particular chosen state.
- **District-wise (/districts)** This is page is also similar to the previous page but it contains lesser number of sections. This is because the database does not have data of tested numbers and vaccination data at a district-level granularity, therefore these stats can not be displayed here.
  - **Select district (/api/districts/list):** First the user needs to select a district. For this they need to select a state first, and then one of its district from the available list of districts of that state. Once selected, they need to press GO button to render the rest of the page.
  - **Summary (/api/districts/summary):** Similar to previous pages, this section deals with aggregate stats in a desired interval of time.
  - **Daily stats (/api/districts/daily):** Similar to previous pages, this section deals with daily values of stats.
- **Admin Panel (/auth and /admin)** When user clicks on the admin panel in the navigation bar they are first taken to the login screen (/auth). This is because the rationale here is that only authorized users must be able to make modifications to the database. Once the user authenticates themselves, they are taken to the admin panel (/admin). The admin panel is first of all divided into 2 tabs, and each tab is further divided into multiple sections.
  - **Update Tab**
    - **Update cases (/api/update/newcases):** In this section the user can update some stats values in the database. User needs to select a state, and one of its corresponding district as well a date for which the modification is meant for. Then the user fills out some fields which correspond to telling the system by what amount should each stat be incremented in the database. It is possible to specify negative values in case the objective is to remove the values from the database. Once done, the user presses the update button. User then receives feedback from the database whether the update was successful or not.
    - **Update vaccinations (/api/update/newvaccinations):** In this section again the user can update some stats values in the database. The difference from the

previous section is that this section deals with stats related to vaccinations only whereas the previous sections deals with COVID related numbers. The second difference is that because the underlying vaccination data is at state-level granularity, the updates must also happen at state level granularity.

  * **Refresh Data (/api/update/refreshall):** In this section there is a single button which the user may press to initiate a full data refresh. By data refresh here we mean the refresh of all materialized views in the database. We will discuss the need of this button in detail in next section of the report but for the time being from user point of view this button needs to pressed about once a day.

- **Management Tab**
  * **New district (/api/management/newdistrit):** In this section user instructs the database to insert a new district into the database. For this they need to select state of which the district would be part of, and a name and population for the district. The rationale behind this section is that as the covid progresses some districts which may never had infections so far may now have the infections, or otherwise it is possible that administration has increased the granularity of data collection. In such cases this section will prove useful.
  * **Update district (/api/management/updatedistrict):** In this section user instructs the database to change name of (an already existing) district.
  * **Remove district (/api/management/deletedistrict):** If at some point it is desired to remove a district for some reason such as if we are no longer collecting data for that district then it is possible to do so with this section. Note that deleting a district means deleting all records in the database corresponding to that district. This however does not imply unwinding the aggregate stats. So for example if delete some district of a state, then no decrement will be made to numbers of that state.

3.2 System view

Couple of lines of explanation for any special functionality that you have. For example: Viewupdate triggers: View V1 materializes 5 star movies from the Movies base table, trigger 'five-star'comes into play whenever a 5-star movie is inserted into 'Movies'. Another example: For securitypurposes, we grant permissions on table 'Movies' only to admins. One more: We built indexes onthe following tables/attributes.

List of all queries that will potentially be fired (many of these will be based on inputs from theuser). This needs to be synchronized with 1. and 2. For each of the item in 1. and 2., write the corresponding SQL query.

3.2.1 Materialized Views

We have used materialized views to improve our performance, as the data would be updated at max 10-20 time in a day, and the queries would be asked in thousands if the site is not popular and the number could go to millions also if the site is popular. So it is better to create materialized view to have a better performance. In particular most of the materialized views we have created are used for data aggregation. In the database tables, for example, we only maintain daily values for a state or a district because this way data update is extremely efficient and maintenance required is low. However more often that not a user is interested in aggregate values of data in a certain period of

time. For example total number of cases so far. All such queries would warrant a full scan over the tables every time to sum the values. Recognizing that this is a common requirement across all queries we have decided to create materialized views which store cumulative value of a parameter till that date.

We have created 8 materialized view, which are described below (SQL file is `app/sql/main.sql`):

1. **India_Vaccine_Daily:** In this view we have summed the daily vaccine related data for all the states on a particular date to get the data of India on that date. So this view contains a row for each date and that row contains vaccine data totals upto that date.

2. **India_Daily:** In this view we have summed the daily cases related data for all the states on a particular date to get the data of India on that date. So this view also contains one row for each date.

3. **India_Cumulative:** In this view we have summed the cumulative cases related data for all the states on a particular date to get the data of India on that date. This view also contains one row for each date.

4. **India_Vaccine_Cumulative:** In this view we have summed the cumulative vaccine related data for all the states on a particular date to get the data of India on that date. This view also contains one row for each date.

5. **State_Cumulative:** In this view we have used the state_daily table to find the cumulative value on that particular date, that is summing the data from all the previous dates. This view contains one row for each pair of state and date.

6. **District_Cumulative:** In this view we have used the district_daily table to find the cumulative value on that particular date, that is summing the data from all the previous dates. This view contains one row for each pair of district and date.

7. **State_Vaccine_cumulative:** In this view we have used the vaccine_daily table to find the cumulative value on that particular date, that is summing the data from all the previous dates. This view contains one row for each pair of state and date.

8. **India_Population:** In this view, we have summed the population of states to find India's total population.

Updating views PostgreSQL does not allow incremental updates to a materialized view. This means that for any update operation on the database if we want to keep all the views upto date then we need to refresh those. However refreshing materialized views at every update is an expensive operation. See the table below for an estimate. Therefore we have decided to strike a compromise. We have provided a refresh button at the front-end to the (authorized-only) users. Once the user (or DBA) is certain that significant updates have had happened to the database the user may press the refresh button to trigger the refresh of materialized views (and regeneration of their corresponding indexes). The rationale here is that small changes to daily values in the database tables will not significantly impact the cumulative values in the views therefore refreshing the views infrequently but regularly such as once per day should prove sufficient for our use. We also use this opportunity to run `ANALYZE` on the database so that the engine is upto to date with current statistics.

Mat. View Name	Time To Refresh (in ms)
state_vaccine_cumulative	140.112
india_population	67.296
india_vaccine_cumulative	122.711
state_cumulative	192.851
district_cumulative	2115.230
india_cumulative	133.638
india_daily	111.423
india_vaccine_daily	133.390

Table 2: Materialized View Refresh Times

3.2.2 Triggers

We are using only one trigger in our database design because of performance reasons (as per our tests, a trigger increased insertion time by 5X).

In our design the trigger we have installed is a trigger on `INSERT/UPDATE` operation on `district_daily` table. Any insert and update operation on this table necessarily corresponds to changing value of one or more stats for a particular district (row). So we have installed the trigger so that these updates are also propagated to the `state_daily` table which stores data in state-wise granularity. The implementation of this trigger is as follows: first we figure out the stateid corresponding to state of this district by querying on `district` table. Then we calculate delta change in the row which has been inserted or updated. We then issue another update query to also update the values in the states table with these delta changes. Here we also take care of the corner case when a row is not already existing in the states table, in that case we simply insert that row. An illustration of this trigger is also shown below.

3.2.3 Indexes

Most of our queries involve selecting a particular date, or a particular state from stateid or a particular district from districtid. Further most of these queries run on materialized views. Since materialized views don't include any index by default we have created some indexes manually over the materialized views for efficient handling of queries.

3.2.4 All Queries

The queries supported by the dashboard are as follows (along with the following API endpoints)

API endpoint	SQL files	Description

/api/india/summary	india_summary.sql	This query finds various statistics like Confirmed Cases, Recovered Cases, Active Cases, Deceased Cases, Other Cases, Tested, Total Vaccine Doses, Active Ratio, Recovery Ratio, Case Fatality Ratio, Test Positivity Ratio, Confirmed per lakh, Recovered per lakh, Active per lakh, Deceased per lakh, Other per lakh, Tested per lakh, Total Vaccine Doses per lakh, Population between given dates.
/api/india/daily	india_daily_daily.sql, india_daily_cumulative.sql, india_daily_avg.sql	This query finds statistics for the last few days (either cumulative, moving 7−day average, daily) like Confirmed Cases, Recovered Cases, Active Cases, Deceased Cases, Other Cases, Tested, Total Vaccine Doses, Active Ratio, Recovery Ratio, Case Fatality Ratio, Test Positivity Ratio
/api/india/vaccine	india_vaccine_summary.sql	This query finds vaccine-related statistics for a given range of dates as follows: Total Dose, First Dose, Second Dose, Males Vaccinated, Females Vaccinated, Transgender Vaccinated, Total Sessions Conducted, Total Covaxin, Total Covishield, Percentage Vaccinated (first dose), Total Dose per lakh, First Dose per lakh, Second Dose per lakh.
/api/india/analysis	analysis_india_daily_daily.sql, analysis_india_daily_cumulative.sql, analysis_india_daily_avg.sql	This query finds the top 3 days between two given dates where a given daily statistic (as 2 rows above) is maximum (either daily, moving 7-day average, or cumulative) is maximum/minimum.
/api/india/liststates	list_state.sql	This query shows the states and their statistics at a glance (Confirmed Cases, Active Cases, Recovered Cases, Deceased Cases, Tested, Vaccinated), sorted according to one of the statistics.
/api/states/list	states_list.sql	This query lists out the states for dropdown purposes
/api/states/summary	state_summary.sql	This query does queries similar to those for /api/india/summary but for a specific state

/api/states/daily /api/states/vaccine	state_daily_daily.sql, state_daily_cumulative.sql, state_daily_avg.sql state_vaccine_summary.sql	This query does queries similar to those for /api/india/daily but for a specific state This query does queries similar to those for /api/india/vaccine but for a specific state
/api/states/analysis	analysis_state_daily_daily.sql, analysis_state_daily_cumulative.sql, analysis_state_daily_avg.sql	This query does queries similar to those for /api/india/analysis but for a specific state
/api/states/listdistricts	list_district.sql	This query shows the districts for a given state at a glance (Confirmed cases, Active Cases, Recovered Cases, Deceased Cases), sorted according to one of the statistics.
/api/districts/list	district_list_statewise.sql, district_list.sql	This query shows a list of districts (either all or for a specific state).
/api/districts/summary	district_summary.sql	This query finds various statistics like Confirmed Cases, Recovered Cases, Active Cases, Deceased Cases, Other Cases, Active Ratio, Recovery Ratio, Case Fatality Ratio, Confirmed per lakh, Recovered per lakh, Active per lakh, Deceased per lakh, Other per lakh, Population, for a given district between two given dates.
/api/districts/daily	district_daily_daily.sql, district_daily_cumulative.sql, district_daily_avg.sql	This query does queries similar to api/states/daily (except the unavailable vaccine data) but for a particular district.
/api/districts/values	district_values.sql	This query prints stats directly from the table and not the materialised views so as to keep the admin view up to date.
/api/states/values	state_values.sql	This query prints stats directly from the table and not the materialised views so as to keep the admin view up to date.
/api/states/vaccinevalues	vaccine_values.sql	This query prints vaccine stats directly from the table and not the materialised views so as to keep the admin view up to date.
/api/management /newdistrict	new_district.sql	This query is to add a new district in case it is not already there.
/api/management /updatedistrict	updated_district.sql	This query is to update a district's name.

/api/management /delete-district	delete_district.sql	This query is to delete a district in case of merging of districts or simple mistakes in creating a district.
/api/update/newcases	insert_cases.sql, update_cases.sql, update_state_cases.sql	This query is to add information about new cases per district in the table.
/api/update /newvaccinations	update_vaccinations.sql, insert_vaccinations.sql	This query is to add information about new vaccinations in the table.

3.3 Query Statistics

List of queries you have tested, and sample run times. You can choose what you would like to reporthere. For example: For Movie Listing: Input by user: 3, Horror, we ran the query on our database andthe results were returned in 2ms. Ideally, this will be a table consisting of the query number (from 3.),parameter values, run times.

The queries supported by the dashboard are as follows (along with the following API endpoints)

API endpoint query	Time (ms)
/api/india/summary?from=2020-01-30&to=2021-03-25	24.06
/api/india/daily?type=Daily&parameter=Confirmed+Cases&ndays=7	25.65
/api/india/vaccine?from=2021-01-16&to=2021-03-25	29.18
/api/india/analysis?from=2021-01-18&to=2021-03-25&type=Daily &parameter=Confirmed+Cases&query=Maximum	23.69
/api/india/liststates?sortedby=Confirmed+Cases &sortedin=Descending	73.70
/api/states/list	13.91
/api/states/summary?stateid=2&from=2020-01-30&to=2021-03-25	33.31
/api/states/daily?stateid=2&type=Daily&parameter=Confirmed+Cases&ndays=7	26.60
/api/states/vaccine?stateid=2&from=2021-01-16&to=2021-03-25	19.10
/api/states/analysis?stateid=2&from=2021-01-18&to=2021-03-25&type=Daily&parameter=Growth+Rate&query=Maximum	22.39
/api/states/listdistricts?stateid=2&sortedby=Confirmed+Cases &sortedin=Descending	596.70
/api/districts/list?stateid=2	11.11
/api/districts/summary?districtid=15&from=2020-01-30&to=2021-03-25	19.84
/api/districts/daily?districtid=15&type=Daily&parameter=Confirmed+Cases &ndays=7	15.21
/api/districts/values?districtid=15&date=2021-04-01	17.49
/api/states/values?stateid=2&date=2021-04-01	13.54
/api/states/vaccinevalues?stateid=2&date=2021-04-01	12.49
/api/management/newdistrict	20.27
/api/management/updatedistrict	17.39
/api/management/deletedistrict	53.04
/api/update/newcases	95.36
/api/update/newvaccinations	13.66

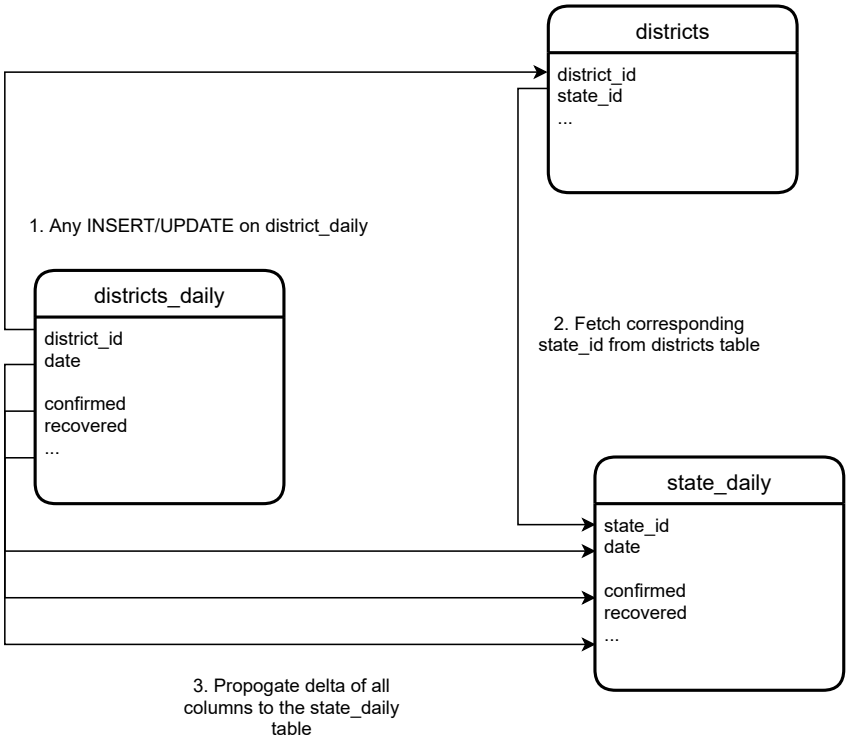


Figure 3: Schematic Illustration of trigger