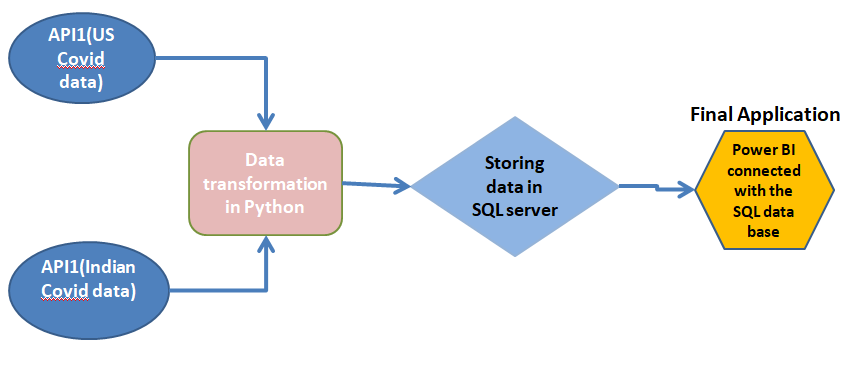
Design Document

A Central Tool to Track the US and India COVID Cases

Tyler(Yue) Huang

<Tel:+15144300566>.



Flow Chart of the Design

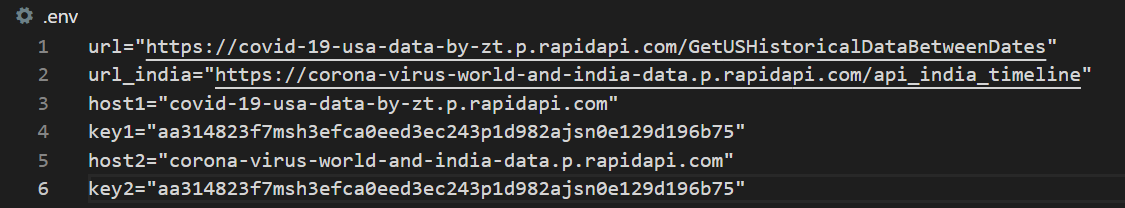
1. **Data source**

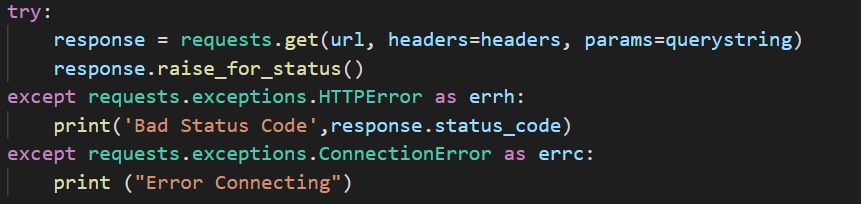
The data is public data from RapidAPI. Two data sources are used. One is US Covid cases data as of early 2020 from API1, and the other is the Indian Covid cases data as of early 2020 from API2. (More data sources can be added as needed)

1. **The process of the design(Code is provided in the Python file)**

As the flow chart indicates

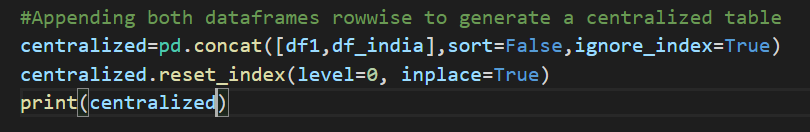
**Step 1: Extracting data from different data sources**

* .env was used to store secret information such as URL and API key. Dot env helps keep the secret info being released, and also facilitates replacement and modification.
* In the data extraction from different APIs, the alerting function was set so we can easily identify HTTP error and connection error.



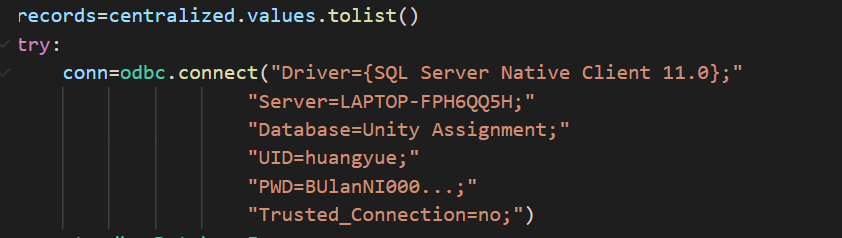
**Step2: Data transformation in python to generate a centralized table**

* After cleaning the data from 2 sources, I unified the format of these two data frames, and appended them row wise to create a centralized table



**Step3: Importing the centralized table to SQL server for a storage purpose**

* Authentication secrets were set



* Inserted the centralized table into the structure in SQL server and wrote a function to avoid duplicates

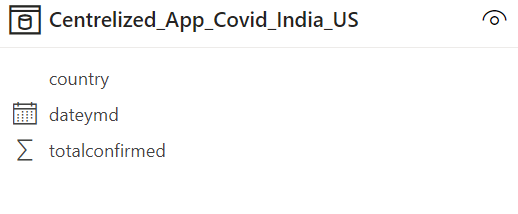


Step4: Connecting SQL server with power BI, this serves as a query and visualization tool

* Created a view in SQL server on the centralized table and connected that view with Power BI(Data folding)

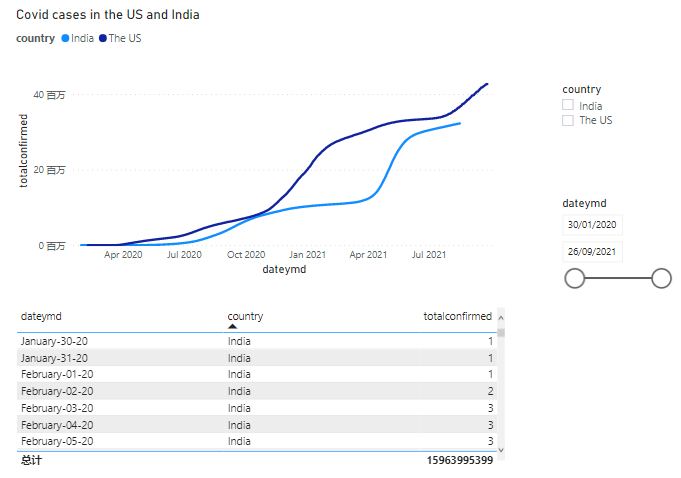
1. **The final tool(power BI)**
2. **Centralized table.**

For the simplicity in this assignment, only 3 fields are kept in the centralized table, *country, dateymd, and totalconfirmed* as the picture below shows, but we can add other fields as needed.



1. **How to query?**

Queries can be done in SQL server or Power BI. In power BI, we use the slicer and filter to do queries in Power BI and export the data after the query. Different measures can be created as needed to generate required visualizations. Below is an example.

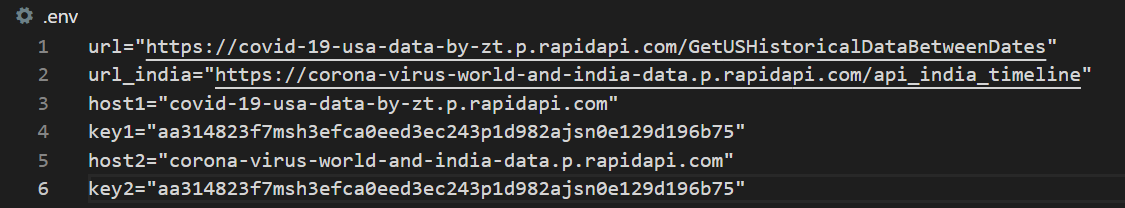


1. **Requirement discussion**
2. **Be able to retrieve data from multiple sources, which may have different APIs per source**

* Two data sources are includes in the design, Covid data USA(API 1) and Covid data India(API2).

We can also add other sources for other countries into the design as needed.

1. **Be able to add support for new data source APIs as simply as possible.**

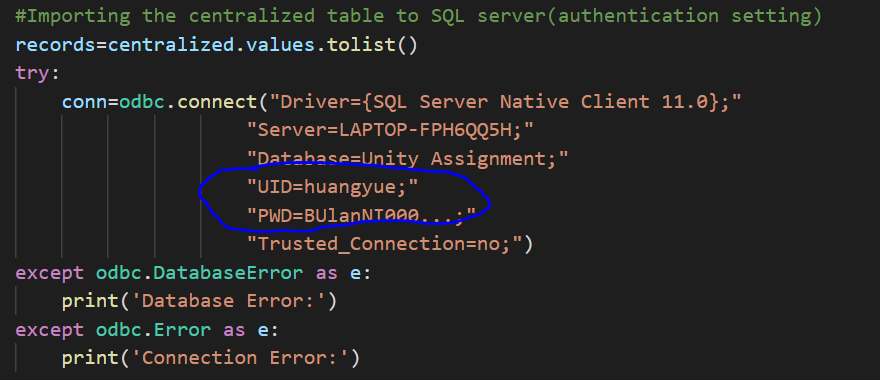
* We can easily add a new data source API. If the format of the new data source is identical as the centralized table, we can append it directly to the centralized table. If the format is different, we need to unify them before appending.
* .env was used to store secret information such as URL and API key. Dot env helps keep the secret info being released, and also facilitates replacement and modification.

1. **Time-series data that is aggregated must be aggregated no more coarsely than daily; more granular where reasonable is preferred.**

* The data is in time-series and is aggregated daily.

1. **The tools to access different data sources must be able to deal with authentication secrets.**

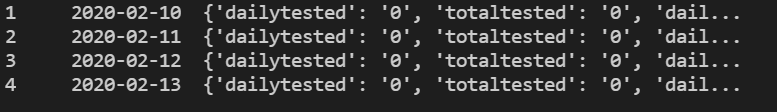
* In the step to import data to SQL server, authentication secrets were set as the screenshot below shows. Only the holder of User id and password can manipulate the data in SQL server.



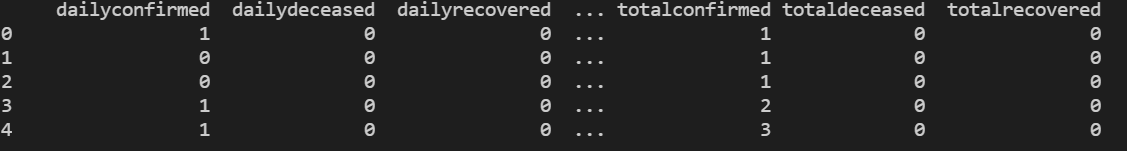
* In power bi, the authorization can be given to the designated members.(Since I know Unity does not use power bi, I do not show it here)

1. **Data to be transformed as needed to be able to be dumped into the centralized data storage in a meaningful way.**

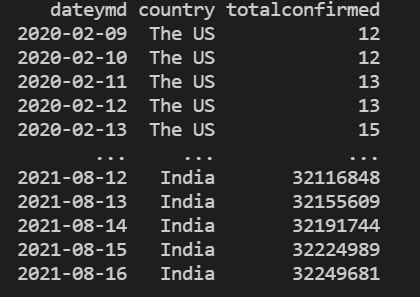
* Blow is the US Covid raw data before being cleaned



* Blow is the India covid raw data before being cleaned

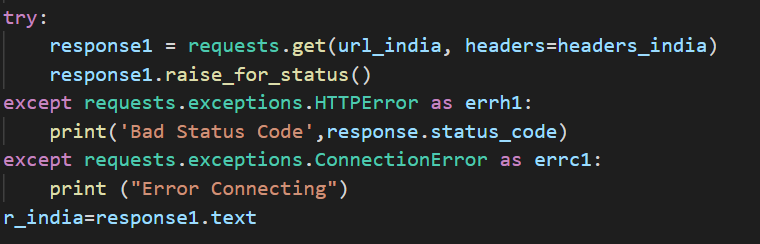


* Because I only need 3 columns: time, country, and total confirmed cases, after data cleaning, I appended the two data sources row wise to generate the centralized table as below. The 3 fields in the centralized table fulfil my present analysis requirement.



1. **Nice-to-Haves Discussion**
2. **Identification and alerting of when a data input source in the service is broken.**

* I incorporated try except function to catch HTTP error and connection error in data extraction from different APIs. When errors happen, python gives me an alert.



1. **Auditability of data in case initial data transformations were not correct, and thus need to be fixed and retroactively updated or re-applied later.**

* We can use Visual Studio SSIS to audit the data transformation. I know the process but since Visual Studio SSIS is not available in 2022 anymore and my laptop cannot install the 2019 version, I might not be able to do it in my laptop. Apology for it!

1. **A solution written using Python or Go is preferred.**

* My solution is written in Python

1. **Avoid from unnecessarily expensive queries.**

* We can use partition and cluster function in big query to avoid expensive queries.