Part-B

Global Database of Covid-19 Vaccinations
Entity Relationship Diagram

Assumptions:

1) iso_code is the only **attribute** which seems to have immense connections amongst all table , provided all the tables have **location** attribute. Except for the **us_state_vaccinations** table as because it contains state-wise locations , not country-wise.

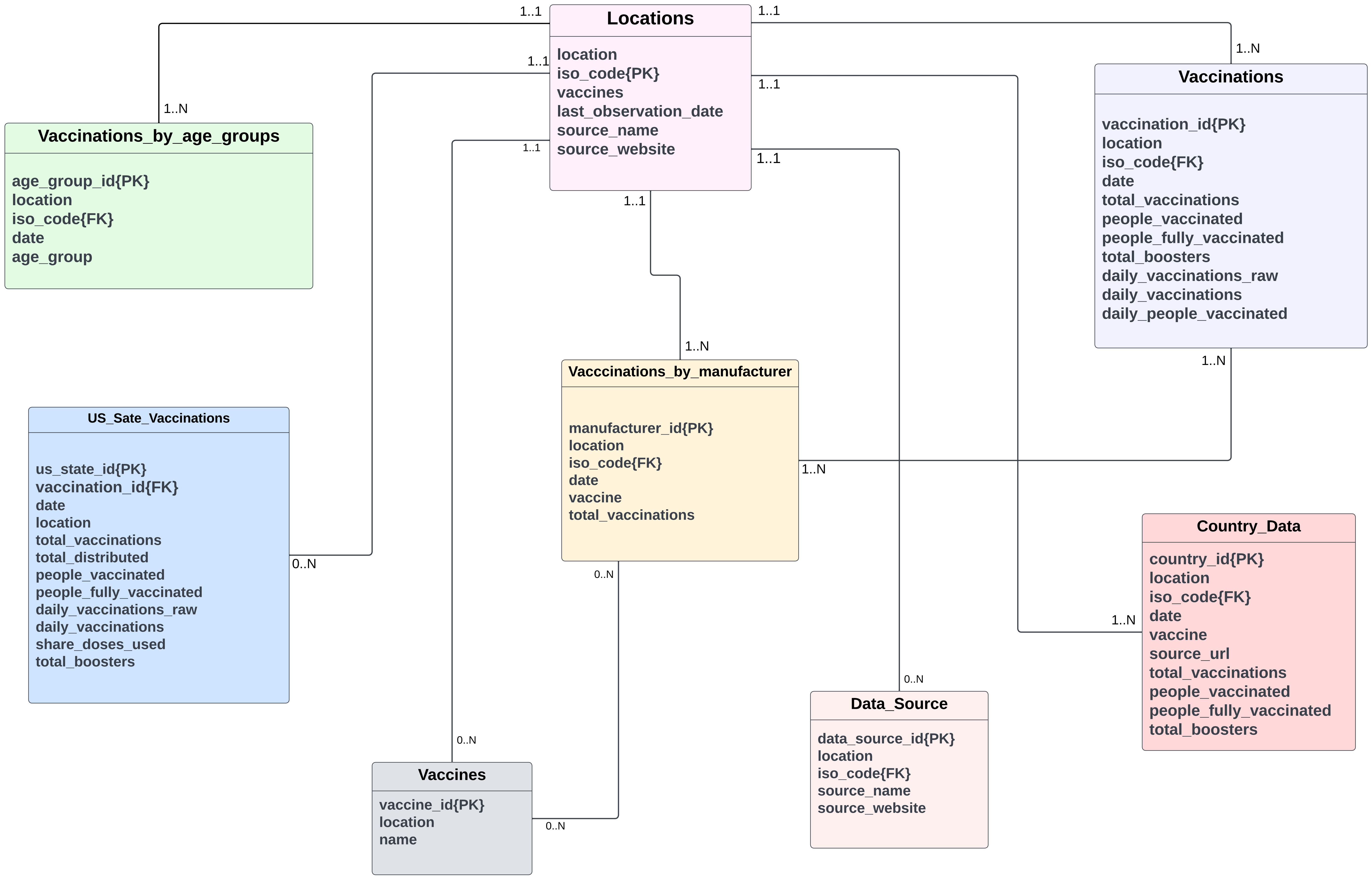
2) A table named **Country_Data** has been created by accomodating all the four countries (i.e Australia , United States , Germany and Italy) into a single table in ER diagram to make it easy to read and understand plus it removes the **data redundancy**.
Under the County_Data table:
i) country_id has been created as a primary key.
ii) iso_code has been considered the foreign key.

3) The **'Date'** format in table **Vaccinations** has been changed from 'dd-mm-yyyy' to 'yyyy-mm-dd' (with the help of MsExcel) in the Database , as the later one is more suitable to perform the queries related to Dates (i.e task-D.5).

4) A table named **Vaccines** has been created with the help of **locations** table , containing: vaccine_id*,location and vaccines. The last column i.e vaccines has been manipulated with the help of MsExcel (Rows to Column feature) and R programming (Gather Function) to further divide each vaccine of each country in **seperate tuple.**

5) A table named **Data_Source** has been created which contains the different data sources used by countries for administering total number of vaccines.
Under the table **Data_Source**:
i) data_source_id has been created as a primary key.
ii) iso_code has been considered as the foreign key.

6) Columns names ends with ***per_hundred*** or ***per_million*** has been removed from the ER diagram as well as the databse , as because those figures were just the dividend values of the existing one. Hence to prevent **data redundancy** , we have removed those figures.



Locations
location
iso_code{PK}
vaccines
last_observation_date
source_name
source_website

Vaccinations_by_age_groups
age_group_id{PK}
location
iso_code{FK}
date
age_group

Vaccinations
vaccination_id{PK}
location
iso_code{FK}
date
total_vaccinations
people_vaccinated
people_fully_vaccinated
total_boosters
daily_vaccinations_raw
daily_vaccinations
daily_people_vaccinated

Vacccinations_by_manufacturer
manufacturer_id{PK}
location
iso_code{FK}
date
vaccine
total_vaccinations

US_Sate_Vaccinations
us_state_id{PK}
vaccination_id{FK}
date
location
total_vaccinations
total_distributed
people_vaccinated
people_fully_vaccinated
daily_vaccinations_raw
daily_vaccinations
share_doses_used
total_boosters

Country_Data
country_id{PK}
location
iso_code{FK}
date
vaccine
source_url
total_vaccinations
people_vaccinated
people_fully_vaccinated
total_boosters

Data_Source
data_source_id{PK}
location
iso_code{FK}
source_name
source_website

Vaccines
vaccine_id{PK}
location
name

Some Notable Assumptions:

i) The **Table Head Names** in the ER diagram are presented in Capital Letters whereas in the actual database in SQL being small letters which are the originals.

ii) The table **locations** share 1..1 and 1..N relationship with almost all the other tables except us_state_vaccinations.

iii) There is 0..N relationship between Manufacturer and Vaccines table.

Normalisation challenges with resulting changes:

Applying the normalization rules

You can apply the data normalization rules (sometimes just called normalization rules) as the next step in your design. You use these rules to see if your tables are structured correctly. The process of applying the rules to your database design is called normalizing the database, or just normalization.

First normal form

First normal form states that at every row and column intersection in the table there, exists a single value, and never a list of values. If you think of each intersection of rows and columns as a cell, each cell can hold only one value.

*Normalisation Challenge-

The Locations table consist the vaccines column which has more than one value in it.

*Resulting Change-

To resolve that , we didn't touched anything with the location as it will create the data untidy , but we did created a new table named vaccine with the help of locations.

Second normal form

Second normal form requires that each non-key column be fully dependent on the entire primary key, not on just part of the key. This rule applies when you have a primary key that consists of more than one column.

- The database created by us doesn't violates the 2NF and hence it accurate.

Third normal form

Third normal form requires that not only every non-key column be dependent on the entire primary key, but that non-key columns be independent of each other.

- In our database , we have make sure that all the keys/columns are independent accept for the tables being created with the help of existing ones. For ex – table vaccine has been created with the help of locations.

Database Schema

location(location,iso_code{PK},vaccines,last_observation_date,source_name,source_website**)**

us_state_vaccinations(us_state_id{PK},vaccination_id{FK},date,location,total_vaccinations,total_distributed,people_vaccinated,people_fully_vaccinated,daily_vaccinations_raw,daily_vaccinations,share_doses_used,total_boosters**)**

vaccinations_by_age_groups(age_group_id{PK},location,iso_code{FK},date,age_group,people_vaccinated_per_hundred,people_fully_vaccinated_per_hundred,people_with_booster_per_hundred**)**

vacccinations_by_manufacturer(manufacturer_id{PK},location,iso_code{FK},date,vaccine,total_vaccinations**)**

vaccinations(vaccination_id{PK},location,iso_code{FK},date,total_vaccinations,people_vaccinated,people_fully_vaccinated,total_boosters,daily_vaccinations_raw,daily_vaccinations,daily_people_vaccinated**)**

country_data(country_id{PK},location,iso_code{FK},date,vaccine,source_url,total_vaccinations,people_vaccinated,people_fully_vaccinated,total_boosters**)**

data_source(data_source_id{PK},location,iso_code{FK},source_name,source_website**)**

vaccines(vaccine_id{PK},location,name**)**