Part B:

Step 1

I first use the bottom-up method(DO Normalization first) to analyze several provided CSV files to have a rough outline of the overall required structure and the relationships between entities.

Locations.csv(iso_code, location_name, vaccines_manufacturer(multi-valued), last_observation_date(LOD), source_name, source_link)

FDs:

iso_code → location_name, LOD, source_name , source_link

vaccines_manufacturer → vaccines_manufacturer (Manufactureres' Name)

Candidate key:

{ **iso_code, vaccines_manufacturer**}$^{+}$={iso_code, location_name, LOD, source_name, source_website, vaccines_manufacturer }

Decomposition:

Merges FDs with the same LHS, then form a relation for each FDs

Locations(iso_code, location_name, LOD, source_link , source_name)

Manufacturers(Mname)

If no relation contains an entire candidate key of the original relation, add one relation containing key

Vaccine_Providers(iso_code*, Mname*)

Vaccinations(location, iso_code, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters, daily_vaccinations_raw, daily_vaccinations, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, daily_vaccinations_per_million, daily_people_vaccinated, daily_people_vaccinated_per_hundred)

Since the original relation schema contains too many attributes that are just for statistical purposes, the following are reasons and steps I simplify some unnecessary attributes:

1. I removed per_hundred attributes because they can be easily calculated by locations' populations and are unnecessary for this database design.
2. I removed daily_vaccinations_raw since the GitHub source recommended that any analysis on daily vaccination rates should be conducted using daily_vaccinations instead.
3. Some locations (countries) didn't report vaccinations regularly, so I will use daily_vaccinations to calculate the monthly total_vaccinations.
4. Already know Iso_code can define location_name, so I removed it directly here to reduce redundants.

After simplify:

Records(iso_code, date, total_vaccinations, daily_vaccinations, daily_people_vaccinated, people_fully_vaccinated, total_boosters)

Candidate key:

{ **iso_code, date** }$^{+}$={iso_code, date, total_vaccinations, daily_vaccinations, daily_people_vaccinated, people_fully_vaccinated, total_boosters }

Step 2 – Business Rule:

After going through all the csv and analyzing them, I attempted to organize the business rules for this project (based on the GitHub README and my own data analysis):

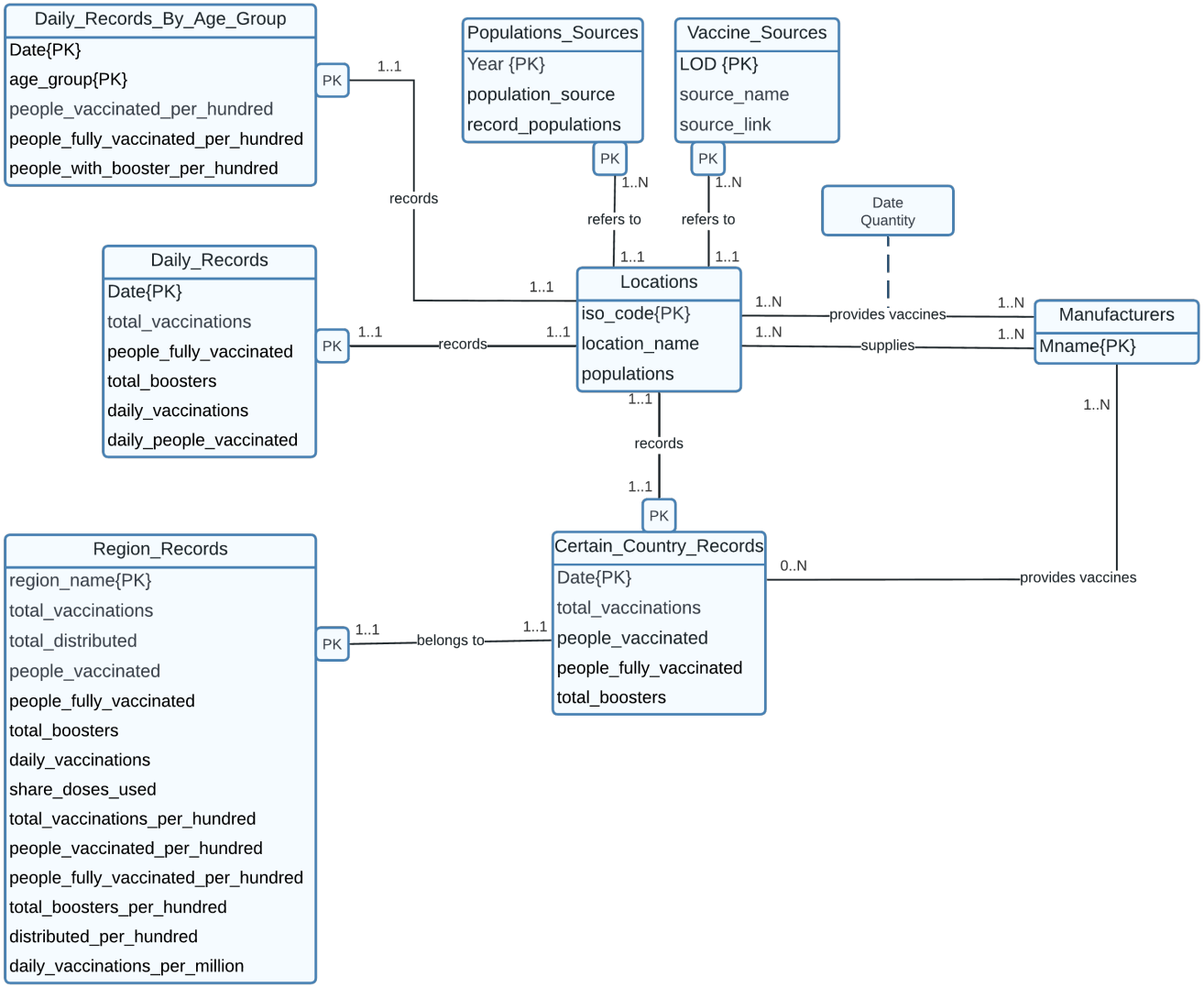
1. **iso_code** can uniquely identify locations (countries, regions, or continents).
2. **last_observation_date (LOD)** is the date of the latest observed data for locations from certain sources.
3. The same **source_name** may have different **source_links**, and the same **source_link** may have different **source_names**.
4. **Year** is the year of the population data obtained from the **population_source** for locations.
5. One manufacturer can provide vaccines to multiple locations, and one location can accept vaccines from multiple manufacturers.
6. Manufacturers may not consistently supply vaccines to certain locations.
7. For some regions, manufacturer supply records are only available for broad areas (e.g., European Union or South Africa), while specific country records will have detailed information on which manufacturer provided vaccines to that location on each day.
8. Here, the assumption is that each location records vaccine administration data daily (see the note below), but it should be noted that the recording time span for each location varies (e.g., some may range from 2020/01 to 2023/12, while others may have ended in 2022/01).
9. Although **vaccinations.csv** includes records for almost all locations, certain countries/locations still have their own records in separate CSV files, which contain different types of data compared to **vaccinations.csv**.
10. Some countries also have detailed data for regions/states within the country (e.g., **us_state_vaccinations.csv**). Since there is currently no detailed population data for these regions, the **per_hundred** statistics in this file should be retained.

Note:

1. **Population** data is additional data introduced (not part of the given dataset for the assignment), which can reduce the need to store various **per_hundred** statistics from the original data. If needed, calculations can be performed using the population data. The data source is **covid-19-data/scripts/input/un/population_latest.csv** from the same GitHub repository.
2. Not every region has accurate daily records of vaccine administration data, but due to the use of the 7-day rolling average smoothing method in the original data, dates with missing data are estimated using data from the previous seven days to derive theoretical data.
3. Some locations (countries) reported **people_fully_vaccinated** data irregularly, so data calculated on a monthly basis might have differences in sampling dates.

4. Due to the 7-day rolling average smoothing method, using **daily_vaccinations** to calculate the monthly **total_vaccinations** may introduce some discrepancies compared to the actual numbers. The same logic applies to **people_vaccinated** and **daily_people_vaccinated**.

Step 3 – ER Diagram:



Some assumptions or notes for ER:

1. The one-to-many relationship between locations and sources is to store the different sources from which locations' data is sampled at different times (e.g., the US population data might be sourced from A in 2022 and from B in 2023). If not indicated, the data is taken from the newest sources by default.
2. The relationship between locations and manufacturers for providing vaccines is recorded on a daily basis. The supplies relationship simply records any instance of supply, marking any manufacturer that has ever supplied vaccines to the location as a supplier.
3. Different types of records (such as by_age_group or certain_country_record) might have different recording periods for the same location, so some corresponding relationships might have a lower limit (participation) of 0.
4. Each location will have only one record per day; there will be no instances of multiple records for the same location on the same day.
5. The **Daily_Records** still retains the original total_vaccinations column to provide accurate actual data when the cumulative total numbers at a certain point in time are needed, rather than summing up the potentially error-prone **daily_vaccinations** data.
6. The records of the same iso in **Daily_Records** and **Daily_Records_By_Age_Group** may have different data collecting timeframes, such as ARG has records data in the whole 2021 in **Daily_Records_By_Age_Group**, but **Daily_Records** only has 2021 January records data. Therefore, I treated them separately. We cannot get **Daily_Records_By_Age_Group** directly from categorizing **Daily_Records** by age group. The date and the number of iso_codes are different in these two csv.

Step 4 – ER Mapping:

Step 1: Map Strong Entities

Locations(iso_code, location_name, populations)

Manufacturers(Mname)

Step 2: Map Weak Entities

Locations(iso_code, location_name, populations)

Manufacturers(Mname)

Vaccine_Sources(iso_code*, LOD, source_name, source_link)

Population_Sources(iso_code*, Year, p_source_link, record_populations)

Daily_Records(iso_code*, date, total_vaccinations, people_fully_vaccinated, total_boosters, daily_vaccinations, daily_people_vaccinated)

Daily_Records_By_Age_Group(iso_code*, date, age_group, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, people_with_booster_per_hundred)

Certain_Country_Records(iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)

Region_Records(iso_code*, date*, region_name, total_vaccinations, total_distributed, people_vaccinated, people_fully_vaccinated, total_boosters, daliy_vaccinations, share_doses_used, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, distributed_per_hundred, daily_vaccinations_per_million)

Step 3: Map 1:1 Relationships

None

Step 4: Map 1:N Relationships

None

Step 5: Map N:N Relationships

Provides_Vaccines(iso_code*, Mname*,Date, Quantity)

Suppliers(iso_code*, Mname*)

Provides_Vaccines_Certain_Country(iso_code*, Mname*,Date*, Sum_Quantity)

Step 6: Multi-valued Attributes

None

Step 7: Map higher-degree relationships

None

Final Schema:

Locations(iso_code, location_name, populations)

Manufacturers(Mname)

Vaccine_Sources(iso_code*, LOD, source_name, source_link)

Population_Sources(iso_code*, Year, p_source_link, record_populations)

Daily_Records(iso_code*, date, total_vaccinations, people_fully_vaccinated, total_boosters, daily_vaccinations, daily_people_vaccinated)

Daily_Records_By_Age_Group(iso_code*, date, age_group, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, people_with_booster_per_hundred)

Certain_Country_Records(iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)

Region_Records(iso_code*, date*, region_name, total_vaccinations, total_distributed, people_vaccinated, people_fully_vaccinated, total_boosters, daliy_vaccinations, share_doses_used, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, distributed_per_hundred, daily_vaccinations_per_million)

Provides_Vaccines(iso_code*, Mname*,Date, Quantity)

Suppliers(iso_code*, Mname*)

Provides_Vaccines_Certain_Country(iso_code*, Mname*,Date*)

Step 5- Check Normal Form:

Locations FDs:

iso_code → location_name, populations

location_name → iso_code, populations

※ Using **iso_code** as the primary key because it is standardized. Although **location_name** is also unique, it may have spelling inconsistencies.

Manufacturers FDs:

Mname → Mname

Vaccine_Sources FDs:

iso_code, LOD → source_name, source_link

Population_Sources FDs:

iso_code, Year → p_source_link, record_populations

Daily_Records FDs:

iso_code, date → total_vaccinations, people_fully_vaccinated, total_boosters, daily_vaccinations, daily_people_vaccinated

Daily_Records_By_Age_Group FDs:

iso_code, date, age_group → people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, people_with_booster_per_hundred

Certain_Country_Records FDs:

iso_code, date → total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters

Region_Records FDs:

iso_code, date, region_name → total_vaccinations, total_distributed, people_vaccinated, people_fully_vaccinated, total_boosters, daliy_vaccinations, share_doses_used, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, distributed_per_hundred, daily_vaccinations_per_million

Provides_Vaccines FDs:

iso_code, Mname, Date → Quantity

Suppliers FDs:

iso_code → iso_code

Mname → Mname

Provides_Vaccines_Certain_Country FDs:

iso_code, Mname, Date → iso_code, Mname, Date

Closures:

Locations: { **iso_code** }$^{+}$= { iso_code, location_name, populations }

Manufacturers: { **Mname** }$^{+}$={ Mname }

Vaccine_Sources: **{ iso_code, LOD }$^{+}$**={ iso_code, LOD, source_name, source_link }

Population_Sources: **{ iso_code, Year }$^{+}$**={ iso_code, Year, p_source_link, record_populations }

Daily_Records: **{ iso_code, date }$^{+}$**={ iso_code, date, total_vaccinations, people_fully_vaccinated, total_boosters, daily_vaccinations, daily_people_vaccinated }

Daily_Records_By_Age_Group: **{ iso_code, date, age_group }**$^{+}$={ iso_code, date, age_group, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, people_with_booster_per_hundred }

Certain_Country_Records: **{ iso_code, date }$^{+}$**={ iso_code, date, t total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters }

Region_Records: **{ iso_code, date, region_name }**+={ iso_code, date, region_name, total_vaccinations, total_distributed, people_vaccinated, people_fully_vaccinated, total_boosters, daliy_vaccinations, share_doses_used, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, distributed_per_hundred, daily_vaccinations_per_million }

Provides_Vaccines: **{ iso_code, Mname, Date }**+={ iso_code, Mname, Date, Quantity }

Suppliers: **{ iso_code, Mname }+**={ iso_code, Mname }

Provides_Vaccines_Certain_Country: **{ iso_code, Mname, Date }+={** iso_code, Mname, Date }

Highest NF:

No Duplicated row/data or multivalued → pass 1NF

No partial dependency on non-prime attributes → pass 2NF

No transitive dependency on non-prime attributes → pass 3NF

Every schema is at 3NF.

Final Schemas:

Locations(iso_code, location_name, populations)

Manufacturers(Mname)

Vaccine_Sources(iso_code*, LOD, source_name, source_link)

Population_Sources(iso_code*, Year, p_source_link, record_populations)

Daily_Records(iso_code*, date, total_vaccinations, people_fully_vaccinated, total_boosters, daily_vaccinations, daily_people_vaccinated)

Daily_Records_By_Age_Group(iso_code*, date, age_group, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, people_with_booster_per_hundred)

Certain_Country_Records(iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)

Region_Records(iso_code*, date*, region_name, total_vaccinations, total_distributed, people_vaccinated, people_fully_vaccinated, total_boosters, daliy_vaccinations, share_doses_used, total_vaccinations_per_hundred, people_vaccinated_per_hundred, people_fully_vaccinated_per_hundred, total_boosters_per_hundred, distributed_per_hundred, daily_vaccinations_per_million)

Provides_Vaccines(iso_code*, Mname*,Date, Quantity)

Suppliers(iso_code*, Mname*)

Provides_Vaccines_Certain_Country(iso_code*, Mname*,Date*)

※ Note for modifying the CSV data:

1. The original **vaccinations_by_manufacturer.csv** (later called **Provides_vaccines.csv**) only has **location_name**. Therefore, I used Excel functions to convert them to the corresponding **iso_codes** in **locations.csv**. Also, in the last section of the table, the **location_name** was listed as European Union, which was not included in the original **locations.csv** and did not have a corresponding **iso_code**.
2. In **vaccinations.csv**, there are additional **iso_codes**: OWID_AFR, OWID_ASI, OWID_EUN, OWID_EUR, OWID_HIC, OWID_LIC, OWID_LMC, OWID_NAM, OWID_OCE, OWID_SAM, OWID_UMC, and OWID_WRL, including the **iso_code** for the European Union. Therefore, I added them to **locations.csv** and also replaced European Union in **Provides_vaccines.csv** with the corresponding **iso_code** to correctly import the data into the database without triggering the Foreign Key Constraint.
3. If specific data is 0 or NULL, it means that the data was not obtained and does not necessarily mean that the value is 0. For example, in the **daily_record**, some locations have data marked as 0 for 2020, which does not mean that there were no vaccinations in that country in 2020, but rather that the database could not collect the vaccination data for that country for 2020.
4. The country records in **Certain_Country_Records** can be stored separately to enhance readability (as shown below), as long as the schema format is not changed. However, when there are more certain countries to store in the future, it will be easier to manage if they are stored together in **Certain_Country_Records**. The main difference between the two methods is the way queries are written.

    - Wales (iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)
    - Canada (iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)
    - United_States (iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)
    - Denmark (iso_code*, date, total_vaccinations, people_vaccinated, people_fully_vaccinated, total_boosters)

5. Due to the lack of detailed information on the number of vaccines provided by each manufacturer in certain regions, In **Provides_Vaccines_Certain_Country**, there is only data on which suppliers provided vaccines daily in that country/location.
6. The original **Vaccinations_By_Age_Group** had many missing values in the age_group column. To successfully import the CSV file, I removed the rows that lacked data in the age_group column.