Databases for Data Science 2023

**Group project documentation**

Group 14

Benjamin Herman 927183

Quang Ngo 1016048

Thinh Nguyen 964010

Erald Shahinas 906845

**Part 1: Database design choice**

In part 2 of the project, our team updated the UML diagram and relational schema that we did in the previous project part according to the provided data. Below are our change notes:

|  |  |
| --- | --- |
| Change notes | Explanation |
| Eliminated the relation Produces(manufacturerID, VaccineID) | Our assumption was that a vaccine could be produced by more than one manufacturer and therefore a relation between Manufacturer and Vaccine is required. According to the given data, that a vaccine ID is given in the table Manufacturer is a better choice. |
| Added name, tempMin and tempMax to Vaccine | We assumed that vaccines have a critical temperature range or maximum and IDs instead of names. According to the given data, the vaccine type table VaccineType (or Vaccine in our case) should have attributes of name, temperature min and temperature max. |
| Added manufacturer to VaccineBatch as a foreign key | We assumed that manufacturers can be retrieved from the relations between the table VaccineBatch, Vaccine and Manufacturer. According to the given data, manufacturers should be added in the table VaccineBatch as a foreign key referenced to the table Manufacturer. |
| Eliminated ID in TransportLog and added (batchID, arrivalDestination, departureDestination) as primary key | We assumed that vaccine batches might go through different transport logs and hence require an ID for each transport log. According to the given data, transport log IDs are not needed and the primary key of the table TransportationLog (or TransportLog in our case) should instead include attributes of batch ID, arrival destination and departure destination. |
| Eliminated type in HealthcareFacility | We assumed that specifying hospitals and clinics could be critically achieved with the attribute type. According to the provided data, this information is not needed in the table VaccinationStations (or HealthcareFacility in our case). |
| Added weekday and worker as primary key of Shift | Our assumption was that each vaccination shift should have an ID. However, according to the given data, the table Shifts (or Shift in our case) should have its primary key to include both weekday and worker. |
| Added hospital to Staff and removed the relation HasStaffMember(shiftID, ssn) | Our assumption was that the information about the workplace of a staff could be retrieved from the relations between the table Staff, Shift and HealthcareFacility. We also assumed that the staff could work for different shifts and hence a relation is required between Staff and Shift. According to the given data, an attribute hospital in the table StaffMembers (or Staff in our case) can indicate the workplace information and the relation HasStaffMember(shiftID, ssn) can be removed. |
| Eliminated eventID and added (vaccinationDate, location) as primary key of VaccinationEvent | We assumed that a vaccination event should have an ID. However, according to the given data, the table Vaccinations (or VaccinationEvent in our case) should have its primary key to include both vaccinationDate and location. |
| Eliminated vaccinationStatus in Patient | Our assumption was that it was important to track patients’ vaccination status to assess whether they are eligible for getting vaccinated at a vaccination event. However, according to the provided data, this information is not needed. |
| Added location to the relation Attends and (vaccinationDate, location, patientSsNo) as primary key | Our assumption was that a relation Attends is needed to connect the class Patient and VaccinationEvent, and IDs are enough to get the other important data. However, according to the provided data, an attribute location is needed in the table VaccinePatients (Attends in our case). As a result, the primary key of this table should include attributes of vaccinationDate, location and patientSsNo. |

After updating the changes above, our database design is as below:

Updated UML

\*Note: Some of table names and attribute names have been modified to match the given data.

Based on our evaluation of the given data, we have put 3 constraints in our database:

|  |  |  |  |
| --- | --- | --- | --- |
| Table | Attribute | Constraint | Note |
| VaccineBatch | dateProduced, expirationDate | Expiration date should be at least 6 months from production date. | Based on our research, all vaccine types in the given data have a shelf life from 6 to 9, 12 or 18 months. |
| TransportationLog | arrivalDate, departureDate | Arrival date should be after or on the same day as departure date. | Simply, arrival date of a vaccine batch cannot be before departure date. |
| Patients | gender | Gender should be either Female (‘F’) or Male (‘M’). | Simply, we ensure that gender data should fall in either female or male.  In other cases, we might need to consider other gender categories such as intersex, transgender, and unknown, which means an additional category ‘Other’ could be included in the gender attribute.  Another note is that PostgreSQL enum can be used in setting a constraint on gender as enum types are useful in defining sets of values that rarely change. |

**Part 2: Results for the required queries**