Project Report Part I: Vaccines

Group 21: Ananov Georgy, Le Duong, Nguyen Tam, Pham Hieu

UML Diagram

The UML diagram can be found at the end of the document.

Relational Schema

Below is the full relational schema of the Vaccine Distribution database that we designed. Further in the document we include a detailed explanation of the roles of each of the relations.

Full Schema

Manufacturer(ID, origin, phone, vaccineID)
VaccineData(vaccineID, nrOfDoses, criticalTemp)
VaccineBatch(batchID, nrOfVaccine, vaccineType, prodDate, expDate)
Produce(manufacturerID, vaccineID)
Supply(batchID, manufacturerID)
Deliver(batchID, facilityName)
MedicalFacility(name, address, phone)
TransportationLog(ID, departureDate, arrivalDate)
Sender(logID, senderName)
Receiver(logID, receiverName)
VaccinationShift(weekday)
StaffMember(ssNo, phone, birthday, vaccinationStatus, role)
Plan(shiftWeekday, facilityName)
WorkOn(staffSSNO, shiftWeekday)
VaccinationEvent(date, location)
Use(batchID, date, location)
Organize(date, location, weekday)
Patient (ssNo, name, birthday, gender, vaccinationStatus)
Symptom (name, critical)
DiagnosedDate (patient, symptom, date)
Attend(date, location, patient)

Explanations

Manufacturer(ID, origin, phone, vaccineID)
VaccineData(vaccineID, nrOfDoses, criticalTemp)
VaccineBatch(batchID, nrOfVaccine, vaccineType, proDate, expDate)
Produce(manufacturerID, vaccineID)
Supply(batchID, manufacturerID)

Deliver(batchID, facilityName)

The **Manufacturer** relation records the data for manufacturers of vaccine, it has attribute *ID* as the ID of the manufacturer, since this ID is unique, we can use it as the key. Furthermore, **Manufacturer** also has the attributes *origin* for the origin country, *phone* to record the phone number of the facility, and *vaccineID* to record the vaccine that the manufacturer produces. The **VaccineData** relation is used to store data about vaccines. It has *vaccineID* as its primary key, recording the unique ID of each vaccine, *nrOfDoses* to store the number of required doses for patients, and *criticalTemp* is the critical temperature of the vaccine. To record data about vaccine batches supplied to medical facilities, we can use the **VaccineBatch** relation. The relation has the *batchID* as its primary key, which records the ID of the batch delivered, *nrOfVaccine* to store how many vaccine doses are transported in the batch, ***vaccineType*** to store the vaccine in that batch, and *prodDate* and *expDate* storing the production and expiry date of the vaccine batch respectively.

For the relational relations of the associations, first we have the **Produce** relation, corresponding to the 'Produce' association between **Manufacturer** and **VaccineData**. It is reasonable to assume that each manufacturer only produces exactly one type of vaccine, while one vaccine can be produced by multiple manufacturers. Thus, this association is a many-one association, and the **Produce** relation will inherit the *manufacturerID* as its primary key from the **Manufacturer** relation and the *vaccineID* from the **VaccineData**. We can say the same for the **Supply** relation, which records the association between the vaccine batches and its manufacturer. One batch is supplied by one manufacturer while one manufacturer can supply multiple vaccine batches, so the relation takes the *batchID* attribute from the **VaccineBatch** as its primary key and the *manufacturerID* attribute from the **Manufacturer** class. Lastly we have the **Deliver** to record the delivery of vaccine batches to medical facilities. As this is also a many-one relationship, the **Deliver** relation will have the *batchID* as its key since it is on the many side, as well as the *facilityName*.

MedicalFacility(name, address, phone)
TransportationLog(ID, departureDate, arrivalDate)
Sender(logID, senderName)
Receiver(logID, receiverName)

The **MedicalFacility** relation contains the data about the medical facilities (the name for both hospital and clinic), including their name, address, phone. Therefore, relation **MedicalFacility** has attributes *name*, *address*, *phone*. The primary key for this relation is the *name* of each facility, as each of them has a unique name.

The **TransportationLog** relation contains information about the transportation log of vaccine batches when they are transported between the hospital and the clinic. The attributes of the relation includes *ID*, *departureDate*, *arrivalDate*. The *ID* of the log is unique, and therefore, it is the primary key for the relation **TransportationLog**.

The **Sender** association between the relation **MedicalFacility** and the relation **TransportationLog** stores the information when the medical facility sends vaccine batches to somewhere else. Since each transportation log can belong to only one facility, and one facility can make multiple transportation, the association is many-one with many on the end

of **MedicalFacility** and one on the end of **TransportationLog**. The **Sender** relation inherits both the *ID* of the **TransportationLog** and the *name* of the **MedicalFacility**, and the key of **Sender** is *logID*.

Similarly, the **Receiver** association stores the information when the medical facility receives vaccine batches from somewhere else. This relation inherits both the *ID* of the **TransportationLog** and the *name* of the **MedicalFacility**, and the key is *logID*.

VaccinationShift(weekday)
StaffMember(ssNo, phone, birthday, vaccinationStatus, role)
Plan(shiftWeekday, facilityName)
WorkOn(staffSSNO, shiftWeekday)

The **VaccinationShift** relation contains the information about the vaccination shift for the workers. Since the shift will be repeated every week, it will only have an attribute *weekday* to indicate the day in the week that the shift happens. This *weekday* will also be the primary key of the relation **VaccinationShift** .

The **StaffMember** relation contains information about the staff. The attributes of the relation include the *ssNo*, *phone*, *birthday*, *vaccinationStatus*, *role*. Attribute *ssNo* of the staff is unique, and therefore, it is the primary key for the relation **StaffMember.** The role can only have value as either “nurse” or “doctor”. The *vaccinationStatus* can only have a value of either 0, 1 or 2 corresponding to the number of doses that the staff takes.

For association, we have the **Plan** relation, corresponding to the “Plan” association between the relation **MedicalFacility** and the relation **VaccinationShift**. The **Plan** relation has attributes *shiftWeekday* (key of relation **MedicalFacility**) and *nameFacility* (key of relation **VaccinationShift**). Since the association between MedicalFacility and VaccinationShift is many-one, the key of **Plan** relation is *shiftWeekday* attribute. The relation **WorkOn** corresponds to the “WorkOn” association between the relation **StaffMember** and the relation **VaccinationShift**. It has two attributes: *staffSSNo* (key of StaffMember) and *shiftWeekday* (key of VaccinationShift). The association is many-many so both these two attributes are the primary keys for relation **WorkOn.**

VaccinationEvent(date, location)
Use(batchID, date, location)
Organize(date, location, weekday)

The central relation of this part of the schema is the **VaccinationEvent** relation. Here we store the date and location (a medical facility) where the vaccination takes place. We then add two associations for the **VaccinationEvent** relation: **Organize** and **Use**.

The **Organize** association describes the connection between vaccination shifts and vaccination events organized during those shifts. Since each shift can organize at most one vaccination event, we only need to include details of the event into the primary key of this association relation.

Next, the **Use** association details which vaccine batch was used during the event. Each batch can be used in at most one event, so we do not need to include details about the event into the primary key for this association relation.

Patient (ssNo, name, birthday, gender, vaccinationStatus)
Symptom (name, critical)
DiagnosedDate (patient, symptom, date)
Attend(date, location, patient)

The **Patient** relation contains the data for every patient, including their social security number, name, date of birth, gender and vaccination status. The primary key for this relation would be the social security number of each patient, as each of them has a unique one and can be used to determine other information. Their symptoms are recorded in the **Symptom** relation. This relation consists of the name of the symptom and the information on whether a particular symptom is critical. For this reason, *critical* would be a value of type Boolean. The primary key for this schema is name, as each symptom has its own name from which its criticality can be inferred. We record patients' diagnosis by the schema **DiagnosedDate**, which contains information about patients (their social security number), the symptom they are diagnosed with (its name), and the day that they are diagnosed. We assume that one patient can be diagnosed with a particular symptom many times during the record, so having *patient* and *symptom* as primary keys are not enough. Thus we also need to have the date that the diagnosis was made as one of the keys. Then the primary keys for **DiagnosedDate** would be *patient, symptom, date.*

We also record each patient's attendance at vaccination events by using the attribute *vaccinationStatus* in the **Patient** relation. This attribute can take values of either 0, 1, or 2, corresponding to the number of doses that they have taken. The data on which patient attends which vaccination events are kept in the **Attend** schema. This relation takes foreign keys *date* and *location* from **VaccinationEvent**, *patient* from **Patient.** Further analysis such as the number of times a patient had a specific vaccine and so on can be deduced from different join-relations and set operations.

Discussion

Functional Dependencies

While recording the Functional Dependencies, we made the following assumptions:

- We assume here that one manufacturer can produce many batches of the same vaccine in a day, so the vaccineType, proDate and expDate are not unique.
- Assume that one manufacturer only produces one vaccine type.
- Assume the one address can have multiple medical facilities (e.g., 1 building can contain different medical facilities).
- We assume that one patient can be diagnosed with a particular symptom many times during the record.

- We assume that the vaccination status of a patient and a staff member can only have a value of either 0, 1 or 2 corresponding to the number of doses that the person takes.
- We also assume that 2 is the maximum number of doses a person can take according to the vaccine (i.e., 1 or 2 doses means the person is fully vaccinated depending on the vaccine type they take).

Below is the list of all non-trivial FDs that are present in the database. We organized them based on their relation of origin for better clarity.

Manufacturer(ID, origin, phoneNr, vaccineID)
- ID → [the rest]
- phone → [the rest] **(potential violation of BCNF)**

VaccineData(vaccineID, nrOfDoses, criticalTemp)
- vaccineID → [the rest]

VaccineBatch(batchID, nrOfVaccine, vaccineType, proDate, expDate)
- batchID → [the rest]

Produce(manufacturerID, vaccineID)
- manufacturerID → vaccineID

Supply(batchID, manufacturerID)
- batchID → manufacturerID

Deliver(batchID, facilityName)
- batchID → facilityName

MedicalFacility(name, address, phone)
- name → [the rest]
- phone → [the rest] **(potential violation of BCNF)**

TransportationLog(ID, departureDate, arrivalDate)
- ID → [the rest]

Sender(logID, senderName)
- logID → senderName

Receiver(logID, receiverName)
- logID → receiverName

StaffMember(ssNo, phone, birthday, vaccinationStatus, role)
- ssNo → [the rest]
- phone → [the rest] **(potential violation of BCNF)**

Plan(shiftWeekday, facilityName)
- shiftWeekday → facilityName

Use(batchID, date, location)
- batchId → [the rest]

Organize(date, location, weekday)
- date, location → weekday

Patient (ssNo, name, birthday, gender, vaccinationStatus):
- ssNo → [the rest]

Symptom (name, critical):
- name → critical

There are five relations that have no non-trivial functional dependencies since all of the attributes in these relations are used as primary keys. These include:

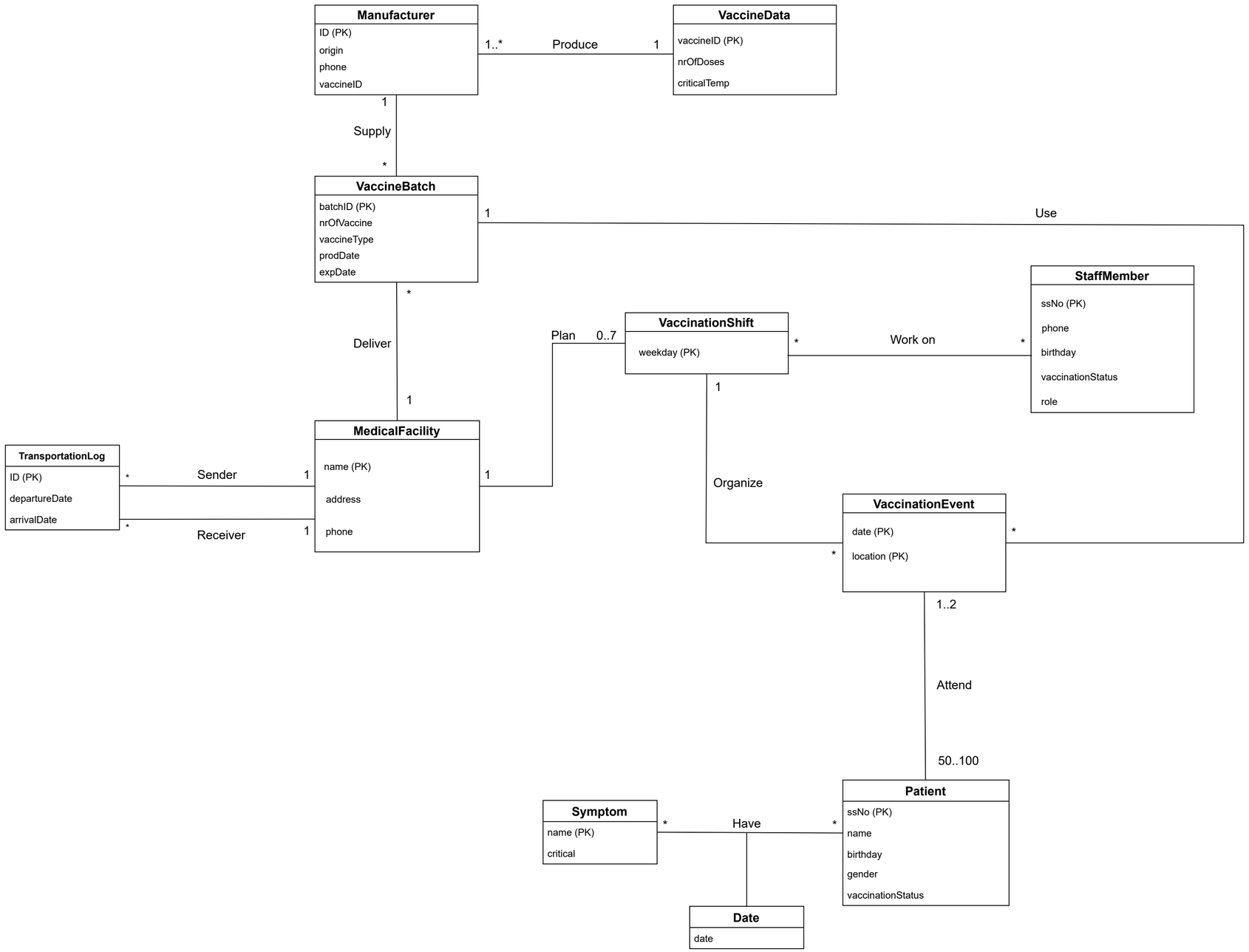
VaccinationShift(weekday)
WorkOn(staffSSNO, shiftWeekday)
VaccinationEvent(date, location)
DiagnosedDate (patient, symptom, date)
Attend(date, location, patient)

BCNF Compliance

Our database mostly follows the Boyce-Codd Normal Form, with one potential exception. For all relations that include phone numbers, there exists an FD of type `phone → [the rest]`, since phone numbers are typically unique. However, this does not violate BCNF, since the phone number is indeed a key of the relation, just not the primary one. Thus, no decomposition is required.

Redundancies

One potential redundancy worth mentioning can be found in the **VaccinationEvent** relation. Here the location (name of the medical facility where the event is held) is stored in the relation and used as a primary key. The redundancy comes from the fact that the same information about the location can also be obtained in another way: **VaccinationEvent** has a one-to-one association with the **VaccinationShift** relation, which in turn has a many-to-one association with the **MedicalFacility** relation. The **MedicalFacility** relation stores the location of the facility, which means that it is possible to obtain the location of a vaccination event without directly looking at the location attribute in the **VaccinaitonEvent** table. However, we decided to accept this redundancy in our database, since eliminating it would make the database schema more complicated and less readable.


Manufacturer
ID (PK)
origin
phone
vaccineID
1..*
Produce
1
VaccineData
vaccineID (PK)
nrOfDoses
criticalTemp
1
Supply
*
VaccineBatch
batchID (PK)
nrOfVaccine
vaccineType
prodDate
expDate
1
Use
*
Deliver
1
Plan
0..7
VaccinationShift
weekday (PK)
*
Work on
*
StaffMember
ssNo (PK)
phone
birthday
vaccinationStatus
role
1
Organize
*
MedicalFacility
name (PK)
address
phone
1
1
TransportationLog
ID (PK)
departureDate
arrivalDate
*
Sender
1
*
Receiver
1
VaccinationEvent
date (PK)
location (PK)
*
*
1..2
Attend
50..100
Patient
ssNo (PK)
name
birthday
gender
vaccinationStatus
Symptom
name (PK)
critical
*
Have
*
Date
date