Data Modeling and Databases I. Project, Phase 3.

Alisa Martyanova, Bekzhan Talgat, Zhandos Kipshakbayev,
Shamil Khastiev, Lev Svalov.
Group - BS18-05.

Table of contents

1. Changes in previous phases.
    a. “What were fixed” paragraph.
    b. Updated schemes.

2. Design decisions.

3. Other related content.
    a. Files for third phase.
    b. **Link to the project in github**.
    c. Short instruction on how to use our application.

Changes in previous phases:

- We decided to change entity **Employee** card to attribute of entity **Employee** as it possibly be a crucial waste of memory and much better to keep such information close to its author(employee).

- We did not store internal communication, because it can be a really large messages per entry, so, that is a waste of memory for our database.

- We decided that our system will have an electronic economic management system, and human as economic manager will keep track of treatment pricing and will take care of agreement issues with patients.

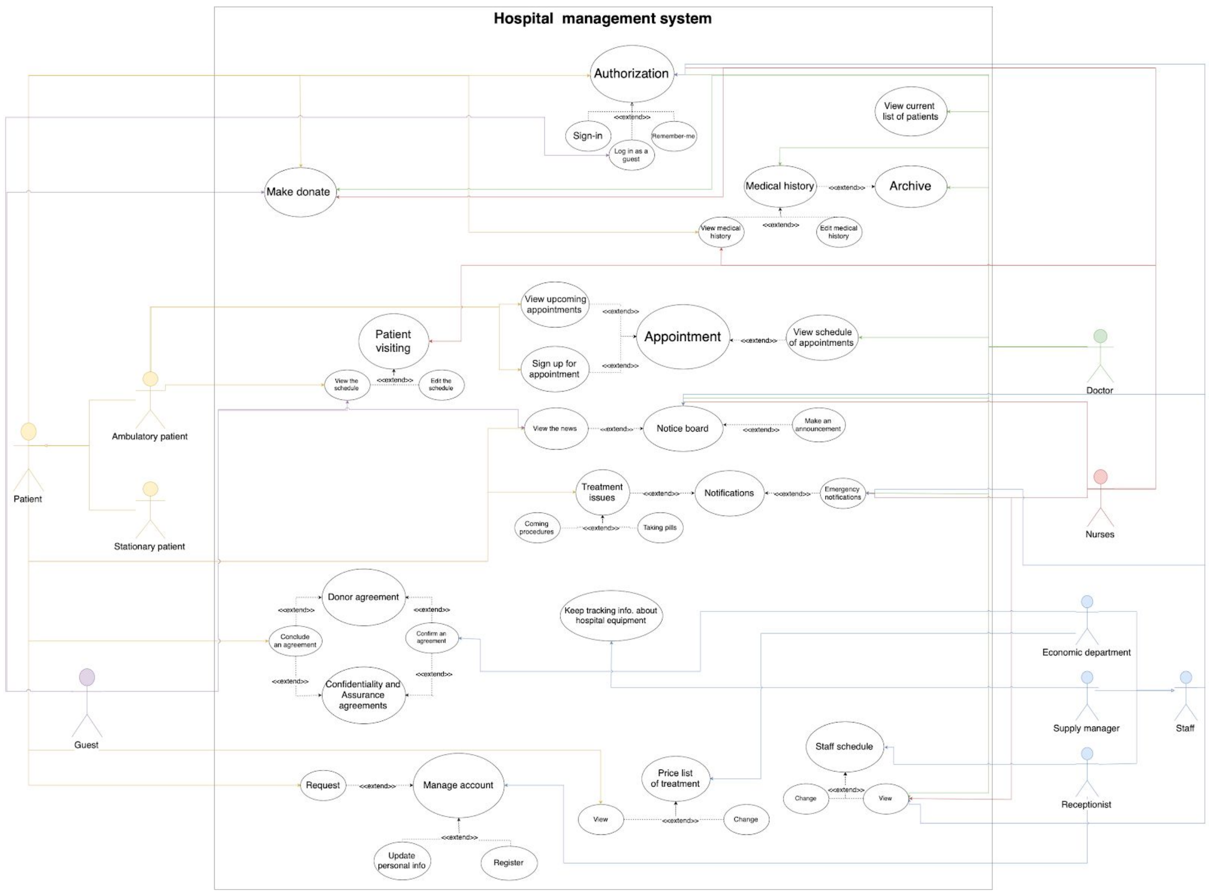
- We decided that to take medical report there is no need of nurse, system will approve medical report for patient according to records in our database.

- We noticed that conclude agreement and donations do the same role as charity and decided to remove charity.

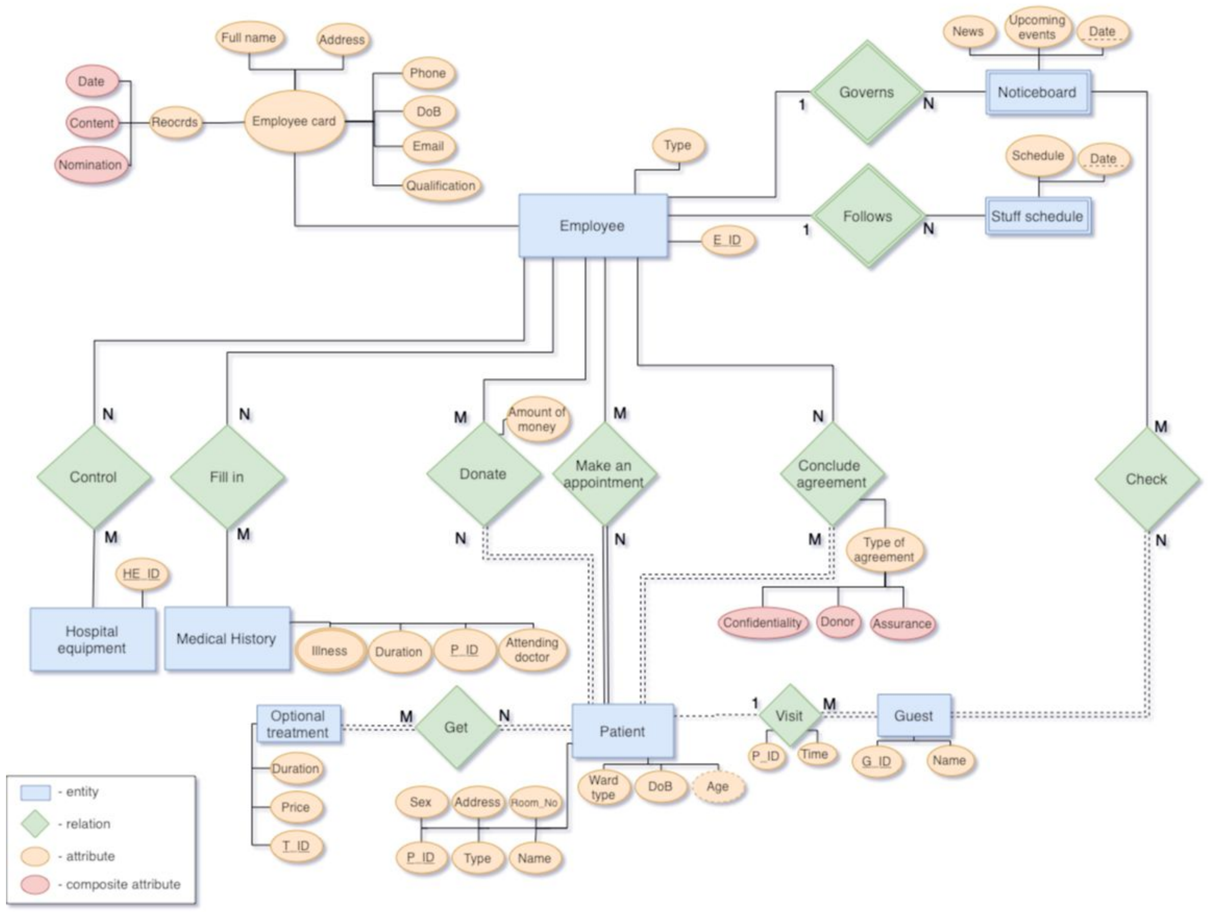
- Canteen menu contains large information per entity, it is again a waste memory for our database. According to this we decided not to implement it in database (it can be accomplished by using different resources).

- All doctors in hospital are qualified enough. We decided that leading list of specialists is not professional.

Use Case diagram:



Entity Relation diagram:



Design Decisions:

Table **Employee:** PRIMARY KEY(E_ID) - for unique identification of employee.

Table **Patient**: PRIMARY KEY(P_ID) - for unique identification of patient.

Table **Stationary_patient**: PRIMARY KEY(P_ID) - It is referenced from Patient, because stationary patient is a type of usual patient.

Table **Guest**: PRIMARY KEY(G_ID) - for unique identification of guest.

Table **Make_an_appointment**: PRIMARY KEY(E_ID, Date) - E_ID is referenced from Employee. By these keys it is possible to identify unique records to doctors at any time slot.

Table **Optional_treatment**: PRIMARY KEY(T_ID) - for unique identification of the treatment.

Table **Notice_board**: PRIMARY KEY(Date,E_ID) - E_ID is referenced from Employee. These keys define who wrote the news and when.

Table **Get_optional_treatment**: PRIMARY KEY(P_ID, T_ID, Date) - every entity of the table can be uniquely identified by the combination of patient received, type of treatment and the date of the treatment applied.

Table **Visit**: PRIMARY KEY(P_ID,G_ID,Date) - Patient is visited by guest once a day so combination of Patient Id, Guest Id and date of visit can uniquely represent visits.

Table **Stuff_schedule**: PRIMARY KEY(Date,E_ID) - every employee has schedule on each day so combination of date and Employee id leads to unique key.

Table **Control**: PRIMARY KEY(HE_ID, supply_manager) - One set of hospital_equipment(HE_ID) can be controlled by one supply_manager(E_ID).

Table **Conclude_agreement**: PRIMARY KEY(Type, P_ID, Date) - one agreement can be uniquely identified by Type of the agreement, Patient(P_ID) who concluded, and Date when he/she did it.

Table **Donate**: PRIMARY_KEY(P_ID, Date) - as type of the Date is timestamp contains also hh:mm:ss it is convenient to keep records of donations uniquely by every patient(P_ID).

Table **Medical_history**: PRIMARY KEY(P_ID, Treatment_start) - simplest way to keep records uniquely of patient's(P_ID) medical history is by date of his treatment_start

Table **Hospital_equipment**: PRIMARY KEY(HE_ID) - for unique identification of a set of hospital_equipment

Other related content:

Files for third phase:

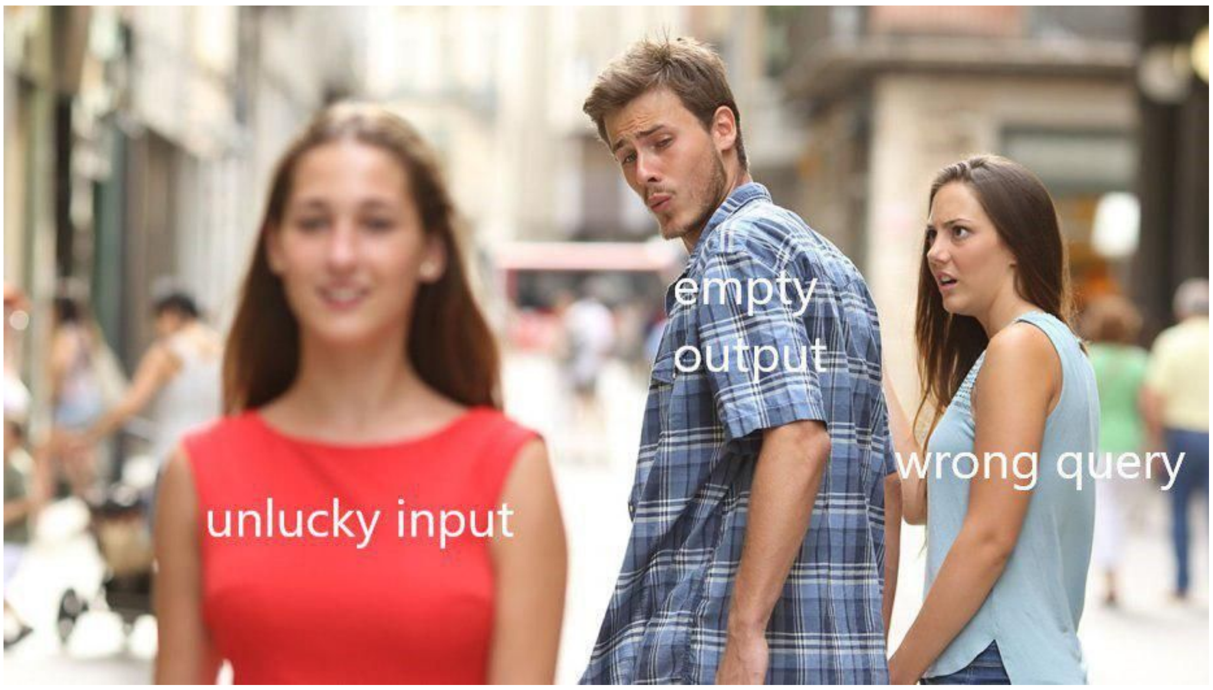
1. database.sql
2. main.py
3. queries.py

Link to the project in github

Short instruction on how to use our application:

1. Open postgresql and create a database, by typing following code line in terminal '''psql -U postgres -d dmd_project -a -f database.sql'''
2. Run main.py
3. Choose whether to generate a new inputs, by typing '1' to terminal (a), or use predefined inputs, by typing any other key.
    a) Consequently input the number of employees, patients, guests, treatments, hospital equipment, and appointments you want to have in the database
4. Wait for data to be generated and inserted into the database
5. Choose number of the query you want to execute from 1 to 5
6. Type '1', if you want to execute other queries, or exit, by typing any other key

EARN POINTS DOING PHASE 3
EARN POINTS FINDING MISTAKES AT 5 AM


unlucky input
empty output
wrong query