Data Bases I
Project "Hospital Database Application"

Paweł Drabczyk

February 15, 2021

Link to the page:
https://damp-reaches-40307.herokuapp.com/
Link to the git repository
https://github.com/Pawel-Drabczyk/Hospital-Database-Application.git

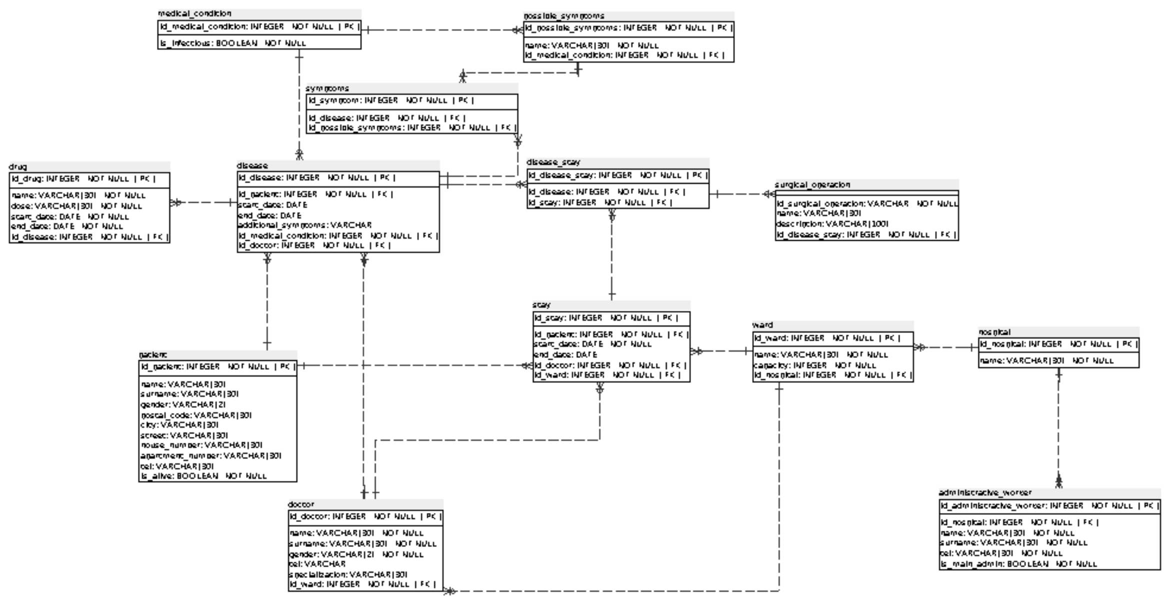
1 Project assumptions, conception

The main goal of the project is to provide a management system for the hospital. The database allows to plan future surgical operations and therapies, monitor patient history and also assign patient to the doctors and hospital wards. The users of the system have different privileges and options:

- The patient - can view the history the treatment history, planned surgical operations and drug therapies
- The doctor - can view the information about all the patients he is assigned to. He can also plan new surgical operation or medications
- The administrative worker - can add, remove and move from one ward to another the doctors and patients. He can also correct mistakes in the treatment history.
- The main administrator - the biggest privileges. Only he can add or remove administrative workers.

Not all of the assumptions have been realised. There is only one privileges level - the level of admin or administrative worker. The dataflow for drug therapies, surgical operations and symptoms is also not implemented.

2 Entity Relation Diagram



medical_condition
id_medical_condition: INTEGER NOT NULL | PK |
is_infectious: BOOLEAN NOT NULL
possible_symptoms
id_possible_symptoms: INTEGER NOT NULL | PK |
name: VARCHAR(30) NOT NULL
id_medical_condition: INTEGER NOT NULL | FK |
symptoms
id_symptom: INTEGER NOT NULL | PK |
id_disease: INTEGER NOT NULL | FK |
id_possible_symptoms: INTEGER NOT NULL | FK |
drug
id_drug: INTEGER NOT NULL | PK |
name: VARCHAR(30) NOT NULL
dose: VARCHAR(30) NOT NULL
start_date: DATE NOT NULL
end_date: DATE NOT NULL
id_disease: INTEGER NOT NULL | FK |
disease
id_disease: INTEGER NOT NULL | PK |
id_patient: INTEGER NOT NULL | FK |
start_date: DATE
end_date: DATE
additional_symptoms: VARCHAR
id_medical_condition: INTEGER NOT NULL | FK |
id_doctor: INTEGER NOT NULL | FK |
disease_stay
id_disease_stay: INTEGER NOT NULL | PK |
id_disease: INTEGER NOT NULL | FK |
id_stay: INTEGER NOT NULL | FK |
surgical_operation
id_surgical_operation: VARCHAR NOT NULL
name: VARCHAR(30)
description: VARCHAR(100)
id_disease_stay: INTEGER NOT NULL | FK |
stay
id_stay: INTEGER NOT NULL | PK |
id_patient: INTEGER NOT NULL | FK |
start_date: DATE NOT NULL
end_date: DATE
id_doctor: INTEGER NOT NULL | FK |
id_ward: INTEGER NOT NULL | FK |
ward
id_ward: INTEGER NOT NULL | PK |
name: VARCHAR(30) NOT NULL
capacity: INTEGER NOT NULL
id_hospital: INTEGER NOT NULL | FK |
hospital
id_hospital: INTEGER NOT NULL | PK |
name: VARCHAR(30) NOT NULL
patient
id_patient: INTEGER NOT NULL | PK |
name: VARCHAR(30)
surname: VARCHAR(30)
gender: VARCHAR(2)
postal_code: VARCHAR(30)
city: VARCHAR(30)
street: VARCHAR(30)
house_number: VARCHAR(30)
apartment_number: VARCHAR(30)
tel: VARCHAR(30)
is_alive: BOOLEAN NOT NULL
doctor
id_doctor: INTEGER NOT NULL | PK |
name: VARCHAR(30) NOT NULL
surname: VARCHAR(30) NOT NULL
gender: VARCHAR(2) NOT NULL
tel: VARCHAR
specialization: VARCHAR(30)
id_ward: INTEGER NOT NULL | FK |
administrative_worker
id_administrative_worker: INTEGER NOT NULL | PK |
id_hospital: INTEGER NOT NULL | FK |
name: VARCHAR(30) NOT NULL
surname: VARCHAR(30) NOT NULL
tel: VARCHAR(30) NOT NULL
is_main_admin: BOOLEAN NOT NULL

Figure 1: Planned Entity Relation Diagram

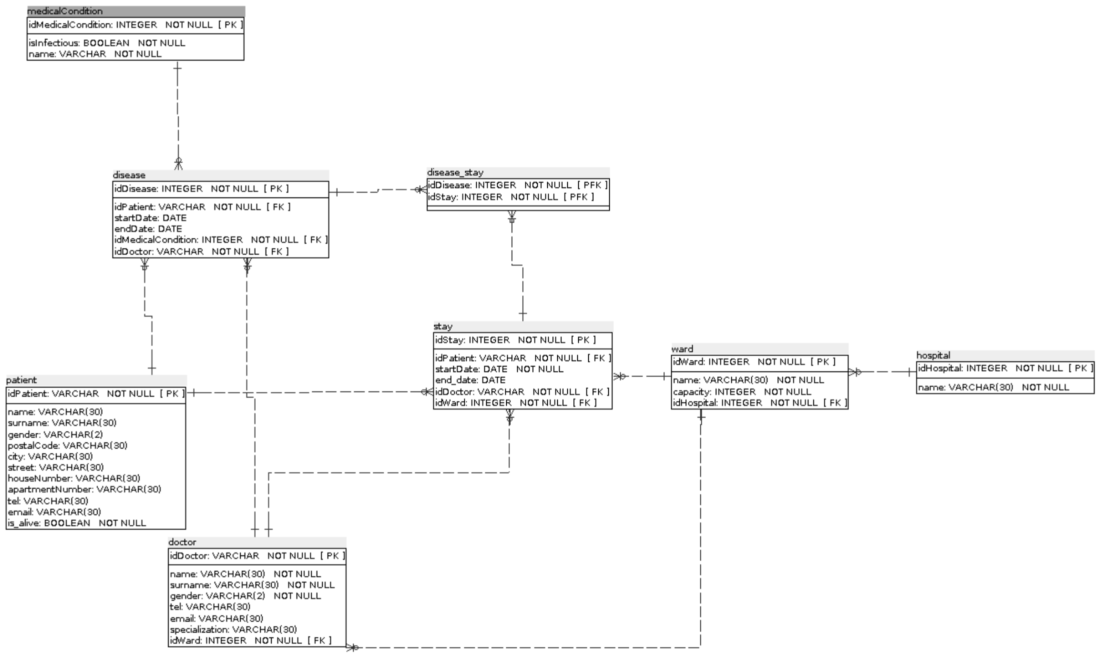

medicalCondition
idMedicalCondition: INTEGER NOT NULL [PK]
isInfectious: BOOLEAN NOT NULL
name: VARCHAR NOT NULL
disease
idDisease: INTEGER NOT NULL [PK]
idPatient: VARCHAR NOT NULL [FK]
startDate: DATE
endDate: DATE
idMedicalCondition: INTEGER NOT NULL [FK]
idDoctor: VARCHAR NOT NULL [FK]
disease_stay
idDisease: INTEGER NOT NULL [PFK]
idStay: INTEGER NOT NULL [PFK]
stay
idStay: INTEGER NOT NULL [PK]
idPatient: VARCHAR NOT NULL [FK]
startDate: DATE NOT NULL
end_date: DATE
idDoctor: VARCHAR NOT NULL [FK]
idWard: INTEGER NOT NULL [FK]
ward
idWard: INTEGER NOT NULL [PK]
name: VARCHAR(30) NOT NULL
capacity: INTEGER NOT NULL
idHospital: INTEGER NOT NULL [FK]
hospital
idHospital: INTEGER NOT NULL [PK]
name: VARCHAR(30) NOT NULL
patient
idPatient: VARCHAR NOT NULL [PK]
name: VARCHAR(30)
surname: VARCHAR(30)
gender: VARCHAR(2)
postalCode: VARCHAR(30)
city: VARCHAR(30)
street: VARCHAR(30)
houseNumber: VARCHAR(30)
apartmentNumber: VARCHAR(30)
tel: VARCHAR(30)
email: VARCHAR(30)
is_alive: BOOLEAN NOT NULL
doctor
idDoctor: VARCHAR NOT NULL [PK]
name: VARCHAR(30) NOT NULL
surname: VARCHAR(30) NOT NULL
gender: VARCHAR(2) NOT NULL
tel: VARCHAR(30)
email: VARCHAR(30)
specialization: VARCHAR(30)
idWard: INTEGER NOT NULL [FK]

Figure 2: Implemented Entity Relation Diagram

3 Logical project

Basing on ERD diagram 2 the data dictionaries have been prepared:

Entity	Variable	Type	Additional Description
Patient	idPatient	VARCHAR(30) NOT NULL	PESEL number, primary key
	name	VARCHAR(30)	
	surname	VARCHAR(30)	
	gender	VARCHAR(2)	
	postalCode	VARCHAR(30)	
	city	VARCHAR(30)	
	Street	VARCHAR(30)	
	houseNumber	VARCHAR(30)	
	apartmentNumber	VARCHAR(30)	
	tel	VARCHAR(30)	
	email	VARCHAR(30)	
	additionalDescription	VARCHAR(30)	
	isAlive	BOOLEAN NOT NULL	

Entity	Variable	Type	Additional Description
Doctor	idDoctor	VARCHAR NOT NULL	PESEL number, PK
	name	VARCHAR(30)	
	surname	VARCHAR(30)	
	gender	VARCHAR(2)	
	tel	VARCHAR(30)	
	email	VARCHAR(30)	
	specialisation	VARCHAR(30)	
	idWard	INTEGER NOT NULL	FK references ward

Entity	Variable	Type	Additional Description
Disease	idDisease	INTEGER NOT NULL	PK
	idPatient	VARCHAR(30) NOT NULL	FK
	startDate	DATE	
	endDate	DATE	
	idMedicalCondition	INTEGER NOT NULL	FK
	idDoctor	VARCHAR NOT NULL	FK

Entity	Variable	Type	Additional Description
Stay	idStay	INTEGER NOT NULL	PK
	idPatient	VARCHAR(30) NOT NULL	FK
	startDate	DATE	
	endDate	DATE	
	idDoctor	VARCHAR NOT NULL	FK
	idWard	INTEGER NOT NULL	FK

Entity	Variable	Type	Additional Information
diseaseStay	idDisease	INTEGER NOT NULL	PFK
	idStay	INTEGER NOT NULL	PFK

Entity	Variable	Type	Additional Information
medicalCondition	idMedicalCondition	INTEGER NOT NULL	PK
	name	VARCHAR NOT NULL	
	isInfectious	BOOLEAN NOT NULL	

Entity	Variable	Type	Additional Information
ward	idWard	INTEGER NOT NULL	PK
	name	VARCHAR(30) NOT NULL	
	capacity	INTEGER NOT NULL	
	idHospital	INTEGER NOT NULL	FK

Definitions of tables can be found in file "hdbapp/DataBase/create.sql".

Proposed database structure is in third normal form, because all non-key values don't depend on other non-key value.

4 Functional Project and Documentation

Each entity has own route. Route leads to 3 forms: first one for inserting the data, the second one for searching for special records and third for updating of the entities.

Forms to fill the tables. Fields for the variables with 'NOT NULL' flag send a notification when they are leaved empty. The effect is achieved with flask's data validators.

Search forms are kind of user friendly. If you leave the field empty, the variable doesn't take part in the selection. Otherwise the conjunction of all conditions is returned. Heroku disallows too long output so don't search for too much objects at once. Used solutions: plpgsql functions ("hdbapp/Database/functions.sql").

Update forms raise the exception and shows the flash communicate if the integrity of the database is in danger. The text of communicate is not always in tune with reality, but it shows that input data are wrong. If you don't specify new primary key the old one is kept.
The navigation bar is located at the top of the page. It is responsible for control over application.

In statistics page there are some simple summaries of wards' capacity, number of different diseases among patients and doctors' occupance. The data are selected using views from file "hdbapp/Database/views.sql".

5 Technical Information

The colors and margins are defined in file "hdbapp/static/main.css".

Directory "hdbapp/templates" consist HTML files. All of the routes extend file "layout.html", which define the constant elements of the page. The templates related to one entity are grouped in directories.

In "hdbapp/Web" there are stored backend functions for the form and also the functions for communication between flask framework and postgreSQL database.

Functions from "hdbapp/views" directory transfer data from the input forms to other functions of the application.

"hdbapp/Database" is a directory with SQL scripts used to create the database. The database was firstly created locally, for the purpose of development. Final version of the application is deployed using "Heroku" service. Files "Procfile" and "requirements" are compulsory for correct deployment.

"__init__.py" is the main file in this application.

References

[1] Requirements to the project
`https://newton.fis.agh.edu.pl/~antek/docs/BD1/BD1_wymagania.pdf`

[2] Requirements to the project documentation
`https://newton.fis.agh.edu.pl/~antek/docs/BD1/BD1_dokumentacja.pdf`