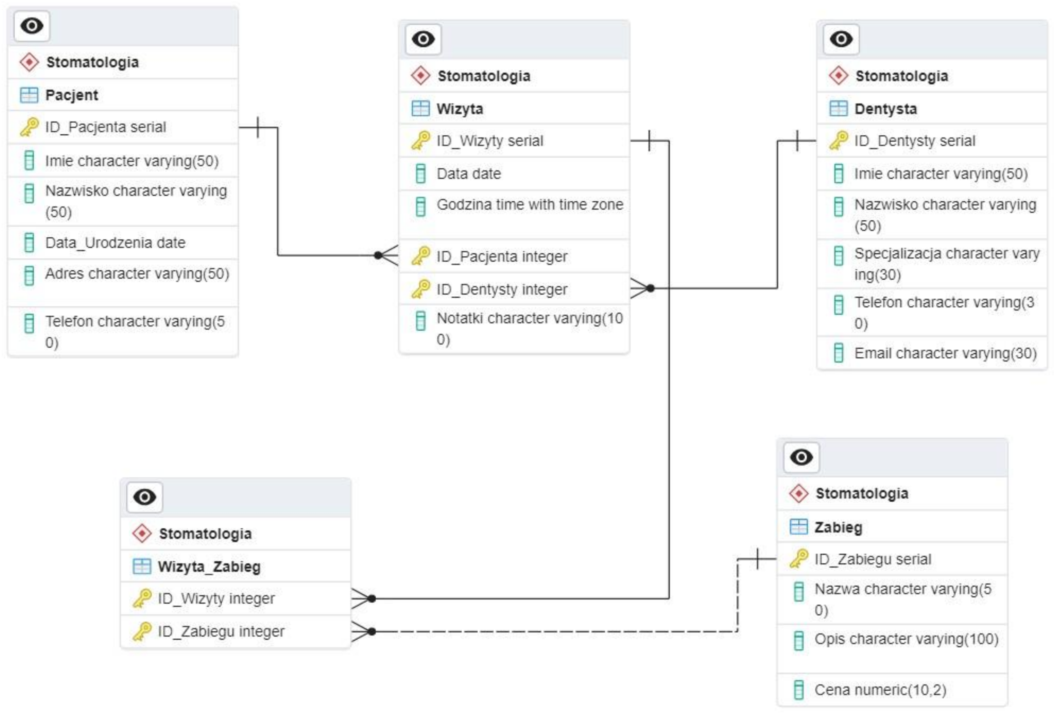
Project Documentation: DentaPro - Dental Clinic Service System

Date: 17.06.2024

1. Project Description

The "DentaPro" system was created to manage patient data and enable the management of dental visits. The application allows adding, updating and deleting patients, managing visits and assigning treatments to visits. The main goal of the project is to improve the work of medical staff through electronic data management.

2. Relational Schemat Diagram


Stomatologia
Pacjent
ID_Pacjenta serial
Imie character varying(50)
Nazwisko character varying (50)
Data_Urodzenia date
Adres character varying(50)
Telefon character varying(50)
Stomatologia
Wizyta
ID_Wizyty serial
Data date
Godzina time with time zone
ID_Pacjenta integer
ID_Dentysty integer
Notatki character varying(100)
Stomatologia
Dentysta
ID_Dentysty serial
Imie character varying(50)
Nazwisko character varying (50)
Specjalizacja character varying(30)
Telefon character varying(30)
Email character varying(30)
Stomatologia
Wizyta_Zabieg
ID_Wizyty integer
ID_Zabiegu integer
Stomatologia
Zabieg
ID_Zabiegu serial
Nazwa character varying(50)
Opis character varying(100)
Cena numeric(10,2)

Patient → Appointment: A patient can have multiple visits, but each visit belongs to a single patient. This is shown as a 1:N relationship between Patient and Appointment.

Dentist → Appointment: A single dentist can perform multiple visits, with each visit performed by a single dentist. This is shown as a 1:N relationship between Dentist and Appointment.

Appointment → Appointment_Procedure: A single visit can consist of multiple procedures (runs), and each procedure can be part of multiple visits. This is reflected in the Appointment_Procedure relationship table, where Appointment_ID is a foreign key to Appointment_ID in the Appointment table.

Procedure → Appointment_Procedure: A run can consist of multiple visits, and each visit can consist of multiple runs. This is also reflected in the Appointment_Procedure relationship table, where Procedure_ID is a foreign key to Procedure_ID in the Procedure table.

Stored procedure

```
CREATE OR REPLACE PROCEDURE stomatologia.dodajpacjenta(
   IN imie TEXT,
   IN nazwisko TEXT,
   IN data_urodzenia DATE,
   IN adres TEXT,
   IN telefon INTEGER,
   IN email TEXT
)
LANGUAGE plpgsql
AS $$
DECLARE
   max_id INTEGER;
BEGIN
   SELECT COALESCE(MAX(id_pacjenta), 0) INTO max_id FROM stomatologia.pacjent;
   INSERT INTO stomatologia.pacjent (id_pacjenta, imię, nazwisko, data_urodzenia, adres, telefon, 
email)
```

```
   VALUES (max_id + 1, imie, nazwisko, data_urodzenia, adres, telefon, email);
END;
$$;

CREATE OR REPLACE PROCEDURE stomatologia.aktualizujdanepacjenta(
   IN new_id_pacjenta INTEGER,
   IN new_imie TEXT,
   IN new_nazwisko TEXT,
   IN new_data_urodzenia DATE,
   IN new_adres TEXT,
   IN new_telefon INTEGER,
   IN new_email TEXT
)
LANGUAGE plpgsql
AS $$
BEGIN
   UPDATE stomatologia.pacjent     SET
imię = new_imie,        nazwisko =
new_nazwisko,        data_urodzenia =
new_data_urodzenia,        adres =
new_adres,        telefon = new_telefon,
email = new_email
   WHERE id_pacjenta = new_id_pacjenta;
END;
$$;

CREATE OR REPLACE PROCEDURE stomatologia.usunpacjenta(
   IN pid INTEGER
)
```

```
LANGUAGE plpgsql
AS $$
DECLARE
    id_to_delete INTEGER;
BEGIN
    SELECT id_pacjenta INTO id_to_delete FROM stomatologia.pacjent WHERE id_pacjenta = pid;
    IF id_to_delete IS NOT NULL THEN
        DELETE FROM stomatologia.pacjent WHERE id_pacjenta = pid;
        UPDATE stomatologia.pacjent
        SET id_pacjenta = id_pacjenta - 1
        WHERE id_pacjenta > pid;
        RAISE NOTICE 'Pacjent o identyfikatorze % został usunięty.', pid;
    ELSE
        RAISE EXCEPTION 'Nie znaleziono pacjenta o podanym ID: %.', pid;
END IF;
END;
$$;
```

Relationship diagram

```
CREATE TABLE stomatologia.pacjent (
id_pacjenta SERIAL PRIMARY KEY,
imię TEXT NOT NULL, nazwisko
TEXT NOT NULL, data_urodzenia
DATE, adres TEXT, telefon TEXT,
email TEXT
);
CREATE TABLE stomatologia.dentysta (
id_dentysty SERIAL PRIMARY KEY,
imię  TEXT  NOT  NULL,          nazwisko
TEXT NOT NULL,
   specjalizacja TEXT,
   telefon VARCHAR,
email TEXT
);
CREATE TABLE stomatologia.wizyta (     id_wizyty SERIAL PRIMARY
KEY,     data DATE NOT NULL,     godzina TIME WITHOUT TIME ZONE,
id_pacjenta INTEGER REFERENCES stomatologia.pacjent(id_pacjenta),
id_dentysty INTEGER REFERENCES stomatologia.dentysta(id_dentysty),
notatki TEXT
);
CREATE  TABLE  stomatologia.zabieg  (
id_zabiegu   SERIAL   PRIMARY   KEY,
nazwa    VARCHAR(100),               opis
VARCHAR(500),                        cena
NUMERIC(10,2)
);

CREATE TABLE stomatologia.wizyta_zabieg (     id_wizyty INTEGER
REFERENCES stomatologia.wizyta(id_wizyty),     id_zabiegu INTEGER
REFERENCES stomatologia.zabieg(id_zabiegu),
   PRIMARY KEY (id_wizyty, id_zabiegu)
);
```

3. Main Application Functions

Adding a Patient

This function allows you to add a new patient to the system, providing their personal data such as name, surname, date of birth, address, phone number and email.

Updating Patient Data

This function allows you to edit the data of an existing patient, including changing personal and contact data.

Deleting a Patient

This function allows you to delete a patient from the system based on their identifier.

Adding a Visit

This function allows you to save a new dental visit in the system, specifying the date, time, patient, dentist and optional notes.

Adding a Procedure

This function allows you to add a new dental procedure to the database, containing the name, description and price of the procedure.

Displaying Visits

This function presents a list of all scheduled visits in the system along with their details.

Assigning Procedures to Visits

This function allows you to assign one or more procedures to a specific visit, which allows you to track the procedures performed during the visit.

Delete visits

Allows you to delete visits by id_visit along with their details.

4. User Manual

The user manual for the "DentaPro" system is available to medical personnel. It contains simple steps for adding, updating and deleting patient data, managing visits and assigning treatments.