Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Базы данных

ОТЧЕТ

по лабораторной работе № 1

на тему: «Разработка требований к схеме данных и

пользовательскому интерфейсу прикладной программы (Больница)»

Выполнил

студент группы 150502: Н.С. Былинский

Проверила

ассистент кафедры ЭВМ: Ю.Ю. Желтко

МИНСК 2024

**1 ИСХОДНЫЕ ДАННЫЕ**

Цель работы:

1. Научиться разрабатывать схему базы данных (БД) и требования к пользовательскому интерфейсу (UI).

2. Ознакомиться с основными командами и интерфейсом работы в PostgreSQL.

Задание:

1. Разработка схемы данных:

    - Спроектировать схему базы данных, включающую основную таблицу, содержащую данные, достаточные для работы пользовательского интерфейса.

    - Добавить 3-4 справочных таблиц (LUT) для заполнения и верификации полей основной таблицы.

    - Не менее двух справочных таблиц должны быть связаны с основной таблицей отношением «многие ко многим».

2. Разработка пользовательского интерфейса:

    - Описать, как пользователи будут взаимодействовать с приложением для работы с базой данных.

- Интерфейс должен поддерживать следующие функции: добавление,

изменение и удаление данных в интерактивном и пакетном режимах.

   3. Технические требования:

    - Разработать спецификацию (техническое задание) для базы данных и пользовательского интерфейса.

- Описание структуры базы данных (какие таблицы нужны, какие поля они содержат).

  - Описание связей между таблицами (какие поля связаны и как).

  - Описание операций, доступных пользователю (что можно добавлять, изменять, удалять).

  - Описание взаимодействия пользователя с интерфейсом программы. <+ макет>

4. Работа с PostgreSQL:

    - Практически освоить команды работы с PostgreSQL для выполнения операций добавления, изменения и удаления данных в базе данных.

**2 ВЫПОЛНЕНИЕ**

## 2.1 Разработка схемы данных

На рисунке 2.1 изображена спроектированная схема базы данных «Больница».

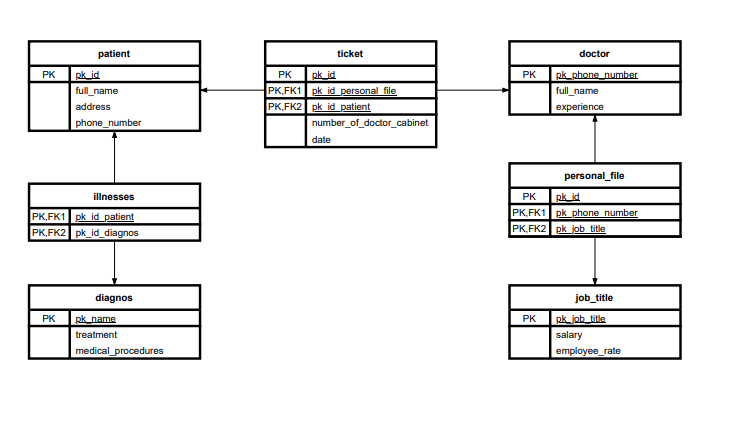


Рисунок 2.1 – uml-диаграмма

**2.1.1 Таблица patient**

Данная таблица содержит информацию о пациентах больницы и имеет следующие поля:

- pk\_id. Данное поле хранит идентификатор каждого пользователя, который автоматически увеличивается при добавлении нового пользователя;

- full\_name. Данное поле содержит Ф.И.О. пациента;

- address. Содержит адрес пациента;

- phone\_number. Содержит номер телефона пациента.

**2.1.2 Таблица ticket**

Эта таблица представляет собой так называемый талон к врачу, которая имеет поля:

- pk\_id. Данное поле является номером талона, которое инкрементируется при добавлении нового талона;

- pk\_phone\_number. Поле, которое ссылается на номер телефона нашего врача в таблице doctor;

- pk\_id\_patient. Поле, ссылающееся на пациента в таблице patient;

- pk\_id\_diagnosis. Поле, в котором указываются названия диагнозов, которые поставил врач по итогу приёма, также ссылается на поле pk\_name в таблице diagnosis;

-number\_of\_doctor\_cabinet. Поле содержит номер кабинета врача;

- date. Поле, указывающее на дату приёма.

**2.1.3 Таблица diagnosis**

Данная таблица хранит информацию о болезнях и их лечении. Данная таблица имеет следующие поля:

- pk\_name. Данное поле хранит название болезни;

- treatment. Это поле хранит подробное описание лечение при определённом заболевании;

- medical\_procedures. Поле хранит рекомендованные процедуры для лечения.

**2.1.4 Таблица doctor**

Эта таблица хранит информацию о врачах больницы. Поля таблицы представлены ниже:

- pk\_phone\_number. Данное поле хранит номер телефона доктора;

- full\_name. Данное поле хранит Ф.И.О врача;

- experience. Данное поле хранит категорию врача.

**2.1.5 Таблица personal\_file**

Данная таблица является с для связи двух таблиц doctor и job\_title и имеет следующие поля:

- pk\_id. Номер личного дела врача;

- pk\_phone\_number. Номер телефона врача;

- pk\_job\_title. Данное поле ссылается на должность врача.

**2.1.6 Таблица job\_title**

Данная таблица хранит информацию о должностях врачей и соответствующем окладе и имеет поля:

- pk\_job\_title. Поле хранит название должностей больницы.

- salary. Поле хранит информацию о зарплате;

- employee\_rate. Данное поле содержит ставку по данной должности.

**2.1.7 Таблица illnesses**

Данная таблица является ссылочной на заболевания пациента.

## 2.2 Разработка пользовательского интерфейса

При разработке пользовательского приложения требуется учитывать лучшие решения UX/UI.

Для разработки будет использоваться язык программирования Java, с использованием фреймворка JavaFX.

Взаимодействие пользователя с приложением будет производиться с помощью клавиатуры и мыши. Приложение будет разделено на пользовательский доступ и административный. Пользовательский доступ предоставляет возможность заказать талон к определённому врачу (см. рисунок 2.1). Административный доступ позволяет удалять заказы или менять их (см. рисунок 2.2).

Будет добавлена поддержка интерактивного и пакетного режимов. Интерактивный режим — это режим, в котором пользователь работает с программой в реальном времени, вводя инструкции вручную и немедленно получая на них отклик. В пакетном режиме команды или запросы собираются в одном файле или пакете и выполняются все вместе, одним запуском, без участия пользователя в реальном времени.

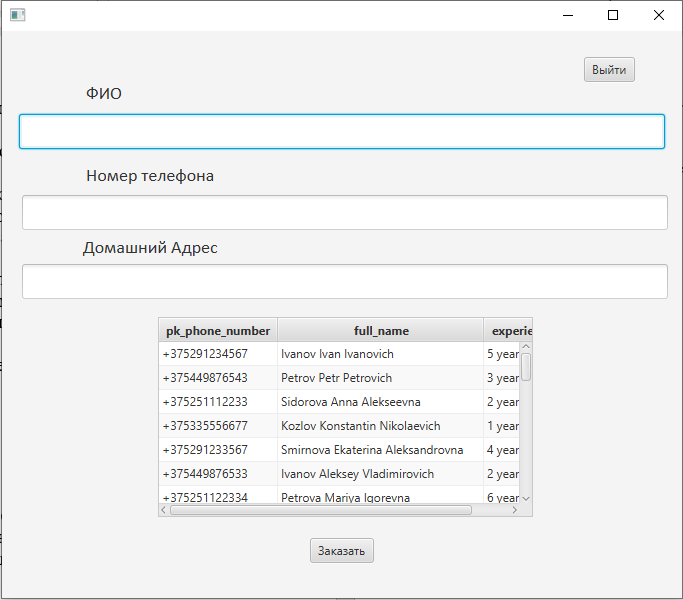


Рисунок 2.1 – Окно для заказа талона

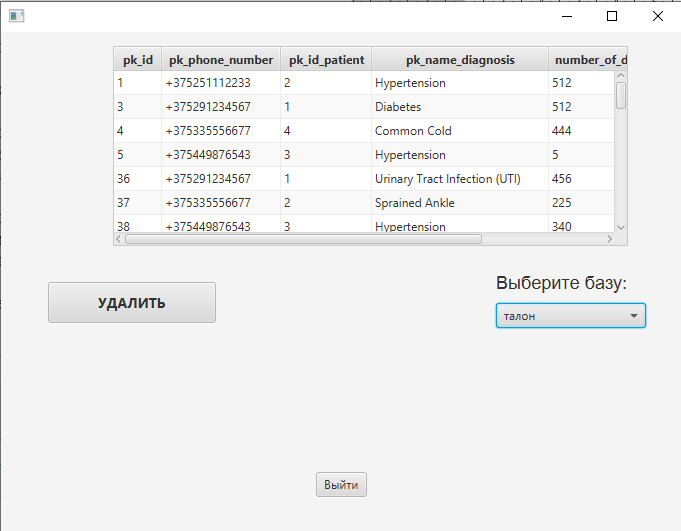


Рисунок 2.2 – Окно для редактирования таблиц

**2.3 Работа с PostgreSQL**

Работа с PostgreSQL включает выполнение различных операций для управления данными в базе данных. Основные команды для создания таблицы «ticket», добавления, удаления и изменения данных с примерами:

1. Создание таблицы

CREATE TABLE ticket

(

pk\_id SERIAL NOT NULL,

pk\_phone\_number varchar(128) NOT NULL REFERENCES doctor(pk\_phone\_number),

pk\_id\_patient integer NOT NULL REFERENCES patient(pk\_id),

pk\_name\_diagnosis varchar(128) NOT NULL REFERENCES diagnosis(pk\_name),

number\_of\_doctor\_cabinet integer NOT NULL,

date varchar(128) NOT NULL,

CONSTRAINT ticket\_pkey PRIMARY KEY(pk\_id,pk\_phone\_number,pk\_id\_patient,pk\_name\_diagnosis)

);

Добавление данных, команда INSERT.

INSERT INTO ticket

VALUES(DEFAULT, '+375291234567',1, 'Urinary Tract Infection (UTI)', 456, '01.03.2024');

Изменение данных, команда UPDATE.

UPDATE ticket

SET pk\_phone\_number = "+375447969930"

WHERE id = 1;

Удаление данных, команда DELETE.

DELETE FROM ticket

WHERE id = 1;

Выборка данных. Чтобы просмотреть данные из таблицы, используется команда SELECT.

SELECT \* FROM ticket

**ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторной работы была создана база данных для больницы. Структура базы данных, содержащая основные и справочные таблицы, позволяет легко отслеживать и обрабатывать данные, гарантируя их целостность и связность.

Пользовательский интерфейс был разработан с акцентом на удобство и интуитивность, что позволяет пользователям быстро выполнять такие операции, как запись на курсы, изменение данных и получение информации. Интерактивный режим обеспечивает мгновенный отклик на действия пользователя, а пакетный режим позволяет эффективно обрабатывать большие объемы данных.

В результате система становится мощным инструментом для управления процессами в автошколе и удобным средством для сотрудников, что способствует улучшению качества и обслуживания пациентов. Реализация этой системы создаст основу для дальнейшего расширения функциональности и оптимизации процессов в автошколе.