**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра ИС**

Курсовая РАБОТА

**по дисциплине «Управление данными»**

**Тема: Разработка и реализация базы данных для работников регистратуры поликлиники.**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8374 |  | Пихтовников К.С. |
| Преподаватель |  | Татарникова Т.М. |

Санкт-Петербург

2020

**ЗАДАНИЕ**

**на курсовую работу**

|  |  |  |
| --- | --- | --- |
| Студент: Пихтовников К. С. | | |
| Группа 8374 | | |
| Тема работы: Разработка и реализация базы данных для работников регистратуры поликлиники. | | |
| Содержание пояснительной записки: «Содержание», «Введение», «Анализ предметной области», «Обоснование модели данных», «Обоснование выбора СУБД», «Описание функций групп пользователей», «Описаний функций управления данными», «Организация защиты БД», «Заключение», «Список использованных источников». | | |
|  | | |
| Предполагаемый объем пояснительной записки:  Не менее 20 страниц. | | |
| Дата выдачи задания: 01.09.2020 | | |
| Дата сдачи работы: 21.12.2020 | | |
| Дата защиты работы: | | |
| Студент гр. 8374 |  | Пихтовников К.С. |
| Преподаватель |  | Татарникова Т.М. |

**Исходные данные**

**Вариант 14.** Спроектировать базу данных, построить программу, обеспечивающую взаимодействие с ней в режиме диалога, для работников ре­гистратуры поликлиники.

В БД должны храниться сведения о больных: ФИО, адрес, диагноз (может быть уточнен), дата заболевания; сведения о врачах: ФИО, № кабинета, № участка, дни и часы приема; описание болез­ней: название (диагноз), симптомы, лекарство. Возможно появле­ние новых больных. Врач может уволиться из поликлиники.

Работникам регистратуры могут потребоваться следующие сведения:

– адрес, дата заболевания, диагноз данного больного,

– ФИО лечащего врача данного больного,

– номер кабинета, дни и часы приема данного врача,

– больные, находящиеся на лечении у данного врача,

– симптомы данного заболевания и рекомендуемое лекарство.

Администратор БД может вносить следующие изменения:

– появление нового больного,

– увольнение врача,

– изменение диагноза.

Необходимо предусмотреть возможность выдачи справки о болезни некоторого больного и отчетов о работе поликлиники (количество больных, ФИО каждого врача и число лечащихся у него больных, количество заболеваний по каждому виду болезни, расписание работы врачей поликлиники).

**Аннотация**

В данной курсовой работе представлена разработка и реализация реляционной базы данных с помощью СУБД MySQL и языка Python. Был сделан анализ предметной области, обоснование модели данных и выбора СУБД MySQL. Также было продемонстрировано тестирование работы программы, обеспечивающую взаимодействие с ней в режиме диалога, для работников регистратуры поликлиники.

**Summary**

This course work presents the development and implementation of a relational database using MySQL and Python. The analysis of the subject area, justification of the data model and the choice of the MySQL DBMS was made. Also, testing of the program, which provides interaction with it in a dialogue mode, was demonstrated for employees of the polyclinic registry.

**Содержание**

[**ВВЕДЕНИЕ** 6](#_Toc59323728)

[**1.** **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ** 7](#_Toc59323729)

[**1.1.** **Анализ предметной области** 7](#_Toc59323730)

[**2.** **ОБОСНОВАНИЕ МОДЕЛИ ДАННЫХ** 9](#_Toc59323731)

[**2.1.** **Основные виды моделей данных** 9](#_Toc59323732)

[**2.2.** **Обоснование выбора модели данных** 9](#_Toc59323733)

[**2.3.** **Создание ER-модели** 10](#_Toc59323734)

[**3.** **ОБОСНОВАНИЕ ВЫБОРА СУБД** 12](#_Toc59323735)

[**3.1.** **Основные виды СУБД** 12](#_Toc59323736)

[**3.2.** **Обоснование выбора СУБД** 12](#_Toc59323737)

[**4.** **ОПИСАНИЕ ФУНКЦИЙ ГРУПП ПОЛЬЗОВАТЕЛЕЙ** 14](#_Toc59323738)

[**4.1.** **Функции групп пользователей** 14](#_Toc59323739)

[**4.2.** **Назначение прав доступа** 14](#_Toc59323740)

[**5.** **ОПИСАНИЕ ФУНКЦИЙ УПРАВЛЕНИЯ ДАННЫМИ** 16](#_Toc59323741)

[**5.1.** **Хранение** 16](#_Toc59323742)

[**5.2.** **Манипулирование** 18](#_Toc59323743)

[**5.3.** **Доступ к данным** 19](#_Toc59323744)

[**5.4.** **Предоставление запрашиваемых данных** 19](#_Toc59323745)

[**6.** **ОРГАНИЗАЦИЯ ЗАЩИТЫ БД** 22](#_Toc59323746)

[**6.1.** **Описание ограниченной целостности** 22](#_Toc59323747)

[**6.2.** **Рекомендуемые средства физической защиты** 23](#_Toc59323748)

[**6.3.** **Процедура подтверждения личности** 24](#_Toc59323749)

[**ЗАКЛЮЧЕНИЕ** 26](#_Toc59323750)

[**список использованных источников** 27](#_Toc59323751)

[**ПРИЛОЖЕНИЕ А. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ БД** 28](#_Toc59323752)

[**ПРИЛОЖЕНИЕ Б. ЛИСТИНГ ПРОГРАММНОГО КОДА** 36](#_Toc59323753)

# **ВВЕДЕНИЕ**

В современном мире в автоматизации предметной области больше всего нуждаются пользователи. Например, такие специалисты как: управленцы, документоведы, технические работники, не являющимся профессиональными программистами. Автоматизация способна фундаментально изменить их подход к профессиональной работе. При этом большое значение имеет дружественный интерфейс для работы пользователя в среде автоматизированной информационной системы.

В предметной области, которая рассматривается в данной курсовой работе, также необходимо провести автоматизацию деятельности. Во-первых, нужно продумать, как и где хранить данные, автоматизировать такие процессы как: поиск и добавление информации, изменение и удаление данных. Например, для нашей предметной области-это поиск лечащего врача больного, запрос адреса и места проживания больного и т.д. Для быстрого получения информации важно учесть такие функции как: выдача справки по запросу работника регистратуры и получение отчета о работе всей поликлиники. Для всего это необходимо создание и проектирование базы данных. В общем случае проектирование базы данных включает в себя: анализ предметной области, обоснование модели данных, выбор СУБД, логическое и физическое проектирование базы данных.

# **АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ**

* 1. **Анализ предметной области**

Предметной областью в данной работе является сведения и данные о врачах и пациентах поликлиники.

Поликлиника - многопрофильное или специализированное лечебно-профилактическое учреждение. Основной задачей поликлиники является оказание квалифицированной специализированной медицинской помощи населению. В поликлинике работают такие основные врачи как: хирург, терапевт, отоларинголог, офтальмолог, травматолог, акушер-гинеколог, уролог, инфекционист, онколог, гастроэнтеролог, кардиолог, эндокринолог и стоматолог. Оказание медицинской помощи пациентам в больнице основывается на двух этапах:

Диагностический – это обнаружение, как можно раньше, изменений физиологических систем организма человека. Обычно на этом этапе пациенту ставится диагноз.

Лечебно-профилактический – это улучшение или восстановление нарушенных физиологических систем организма. На этом этапе врач определяет лечение.

Пациент-это физическое лицо, которому оказывается медицинские услуги. Врач сам определяет для больного необходимый перечень лечебных мероприятий в поликлинике.

Программа, которую предстоит разработать в рамках данной курсовой работы, должна будет являться помощником для пользователя, а именно для работника регистратуры и администратора базы данных.

Работнику регистратуры поликлиники может понадобиться информация о больных, врачах, известных заболеваниях, симптомах, лекарствах, а также справка о заболеваниях пациента и отчет о работе поликлиники.

Необходимо предусмотреть, чтобы в базе данных хранились записи о больных, врачах, диагнозах каждого больного, описания всех известных диагнозов, а также их симптомы и рекомендуемое лекарство. Также должны храниться сведения об участках, за какими адресами закреплены и какими врачами обслуживаются.

При выводе информации можно узнать следующие сведения:

* адрес больного и его заболевания;
* лечащие врачи заданного больного;
* симптомы заданного заболевания и рекомендуемое лекарство;
* время работы врача и участок, который им обслуживается;
* пациенты заданного врача;

Администратор базы данных обладает следующими функциями:

* добавлять нового больного;
* удалять больного из базы данных;
* добавлять и изменять участок, номер кабинета и время приема врача;
* увольнять врача;
* создавать нового пользователя;
* удалять пользователя из базы данных

В базе данных работник регистратуры может заказать выдачу справки о болезни пациента, а также просмотреть отчет о работе всей поликлиники.

Именно такое представление информации позволяет более комфортнее и эргономичнее использовать базу данных. В этом случае вводимая информация может быть сведена к выбору целесообразных материалов и источников из списка или таблицы, что несомненно, существенно усовершенствует скорость ввода данных и послужит поддержкой во избежание неверного ввода параметров.

# **ОБОСНОВАНИЕ МОДЕЛИ ДАННЫХ**

## **Основные виды моделей данных**

Основой любой базы данных является модель данных. Модель данных - это различная совокупность структур данных, ограничений целостности и операций их обработки. Выбор модели данных зависит от поставленной задачи и анализа информации предметной области. Современная классификация СУБД предусматривает реализацию моделей двух групп: модели структурированных и модели неструктурированных данных. К первой группе относятся иерархические, сетевые и реляционные модели данных, а вторую группу образуют модели неструктурированных данных (нереляционные модели) - базы данных NoSQL.

Структурированные данные- это данные любых типов сведений, которые находятся в фиксированном поле, в записи или файле. Они содержат в себе материалы, содержащиеся в реляционных базах данных и электронных таблицах. Работа с этими данными происходит с помощью такого языка запросов, как SQL.

Неструктурированные данные- это те данные, которые не могут быть легко классифицированы и организованны заранее определенным образом. К ним относятся фотографии, электронные письма, файлы PDF, документы для обработки текстов и др.

## **Обоснование выбора модели данных**

Наша текущая база данных, прежде всего, позволяет работникам регистратуры поликлиники упростить выполнение своей работы, а также повысить ее эффективность за счет автоматизации определенных действий. База данных позволяет систематизировать информацию, пользоваться функциями ее быстрого поиска. Отсюда следует, что работники регистратуры могут не тратить время на бумажные работы. Не искать в документах какую-либо информацию, перебирая листы руками. С помощью базы данных они могут брать различные справки о пациентах, смотреть отчет о работе всей поликлинике и предоставлять его начальству.

Структура проанализированной информации позволяет сказать нам о том, что наиболее подходящей для проектирования базы данных будет реляционная модель данных. Достоинство этой модели заключается в простоте, понятности и удобности ее физической реализации. Именно простота и удобность использования делают эту модель одной из самых популярных и самых используемых на сегодняшний момент. Реляционная модель данных способна обеспечит целостность данных при вставке, удалении и изменении записей. Она позволяет организовать все виды связей: один к одному, один ко многим, многие ко многим. К недостаткам традиционных реляционных моделей данных можно отнести то, что при большом объеме базы данных увеличивается время доступа и объем служебной информации.

* 1. **Создание ER-модели**

Для реализации нашей базы данных нужно установить все связи между объектами.

Рассмотрим следующие сущности:

* Больной (patients)
* Врач (doctors)
* Диагноз ( diagnos )
* Диагноз больного ( diagnosis\_of\_the\_patient )
* Описание болезней ( description\_disease )
* Рекомендуемые лекарства (medicaments)
* Адрес-участок ( adress\_area )
* Район ( area )

Сущности adress\_area и area позволяют реализовать отношения M:M ( многие ко многим).

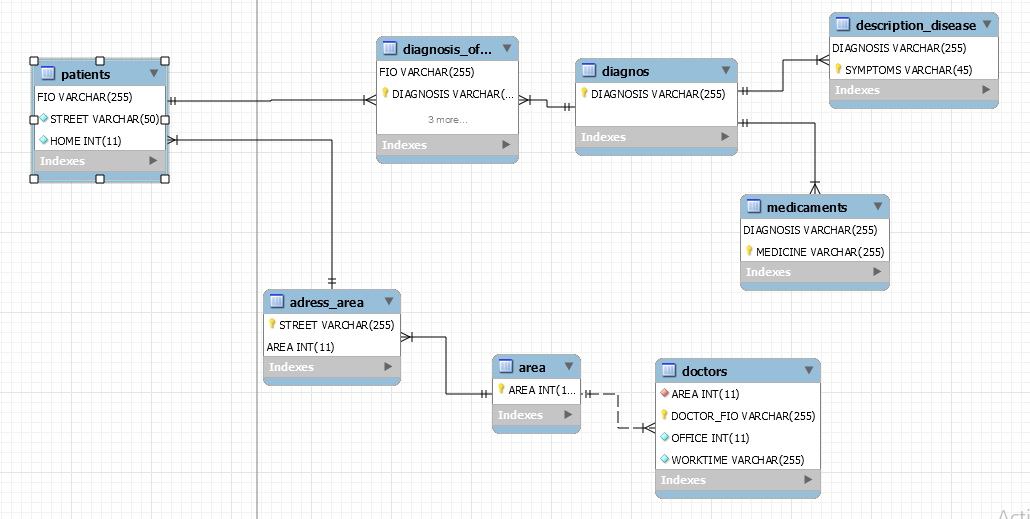
Сущность patients и diagnosis\_of\_the\_patient имеют свзять 1:M ( один ко многим ).

Сущность diagnos имеет связь 1:M (один ко многим) с diagnosis\_of\_the\_patient, description\_disease и medicaments.

Сущность adress\_area имеет связь 1:M (один ко многим) с patients.

Сущность area имеет связь 1:M (один ко многим) с adress\_area и doctors.

На основе этих сущностей получена следующая ER-модель (схема данных)

****

# **ОБОСНОВАНИЕ ВЫБОРА СУБД**

## **Основные виды СУБД**

Система управления базами данных (СУБД) — совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных. Как уже было ранее сказано, современные СУБД предусматривают реализацию моделей структурированных и неструктурированных данных. Например, к первой группе относятся такие СУБД, как: MySQL, PostrgreSQL, SQLite, ко второй: DynamoDB, Riak, MongoDB, Apache Cassandra. По степени распределенности различают локальные и распределенные СУБД, а по способу доступа к БД: файл-серверные, клиент-серверные и встраиваемые.

## **Обоснование выбора СУБД**

В данной курсовой работе используются структурированные данные, поэтому был сделан выбор в пользу реляционной модели СУБД MySQL.

MySQL — свободная реляционная система управления базами данных. СУБД MySQL характеризуется хорошей скоростью работы, гибкостью и надежностью. Она является клиент-серверной СУБД и свободно распространяемым кросс-платформенным продуктом. Также MySQL имеет двойное лицензирование и может распространяться в соответствии с условиями лицензии GPL. MySQL портирована на большое количество таких платформ, как: Linux, Mac OS X, Windows 95, Windows 98, Windows Server 2003, WinCE, Windows 7 и др. Она имеет API для таких языков, как: Delphi, C, C++, Java, Perl, PHP, Python, библиотеки для языков платформы .NET.

Одними из главных плюсов MySQL является высокая скорость обработки информации и ее функциональность. MySQL поддерживает практически всю часть функционала SQL, а также в нее встроено много функций безопасности. MySQL может работать с действительно большими объёмами данных, и походит для масштабируемых приложений. Данная СУБД способна поддерживать многопоточность (несколько одновременных запросов), оптимизацию связей с присоединением многих данных за один проход, гибкую систему привилегий и паролей.

# **ОПИСАНИЕ ФУНКЦИЙ ГРУПП ПОЛЬЗОВАТЕЛЕЙ**

## **Функции групп пользователей**

В качестве групп пользователей были выбраны администраторы и обычные

пользователи-работники регистратуры. Все данные могут изменяться только администратором, а пользователи имеют доступ к просмотру информации.

Функции для администратора БД:

1. Добавление нового пациента (INSERT)
2. Удаление пациента (DELETE)
3. Добавление нового врача (INSERT)
4. Увольнение (удаление) врача (DELETE)
5. Изменение диагноза (UPDATE)
6. Создание (регистрация) нового пользователя (CREATE USER)
7. Удаление пользователя (DROP USER)

Функции для пользователя (работников регистратуры):

1. Получение адреса, даты заболевания, диагноз больного (SELECT)
2. Получение ФИО лечащего врача больного (SELECT)
3. Узнать номер кабинета, дни и часы приема данного врача (SELECT)
4. Какие больные находятся у данного врача (SELECT)
5. Узнать симптомы заболевания и лечение (SELECT)
6. Получить справку (SELECT)
7. Получить отчет (SELECT)

## **Назначение прав доступа**

К первой группе нашей таблице относится администратор, ко второй- пользователь.

Администратору доступны функции – чтение, добавление, модификация и удаление данных для всех таблиц, а также функция создания и удаления пользователя, распределение прав доступа. Пользователю же доступно только чтение данных.

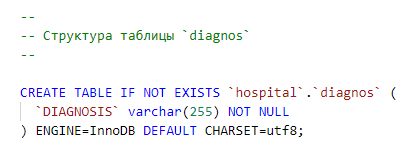
|  |  |  |
| --- | --- | --- |
| Объект | Администратор | Пользователь |
| address\_area | SUID | S |
| area | SUID | S |
| description\_disease | SUID | S |
| diagnos | SUID | S |
| diagnosis\_of\_the\_patient | SUID | S |
| doctors | SUID | S |
| medicaments | SUID | S |
| patients | SUID | S |
| user | SUID, create user, GRANT | - |

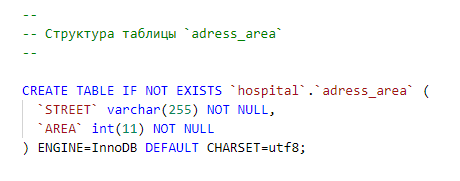
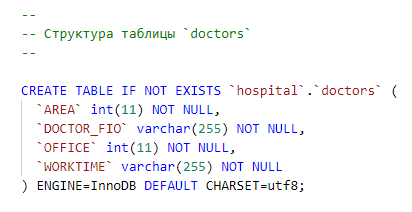
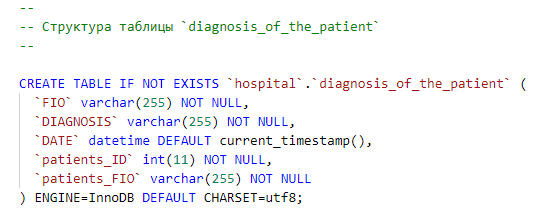
*Табл. 1. Распределение прав*

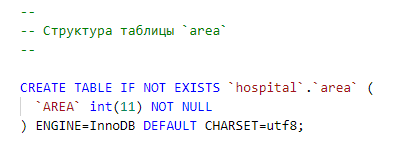
# **ОПИСАНИЕ ФУНКЦИЙ УПРАВЛЕНИЯ ДАННЫМИ**

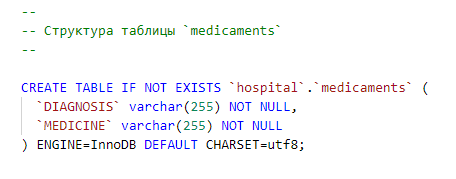
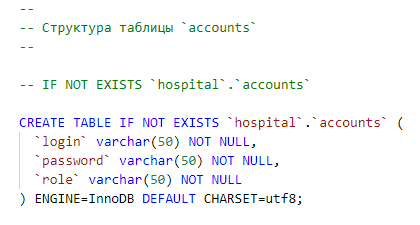
## **Хранение**

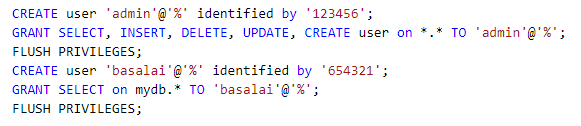
Примеры создания информационных объектов:

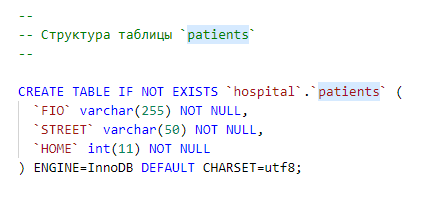






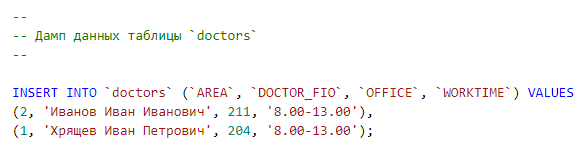


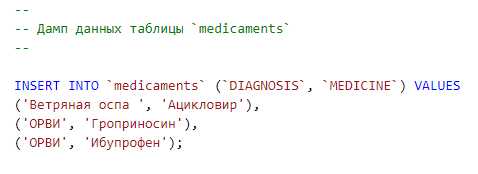
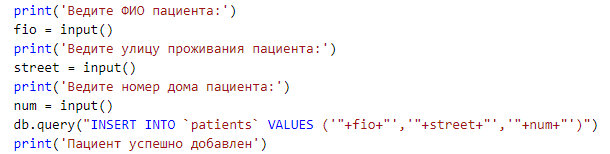




## **Манипулирование**

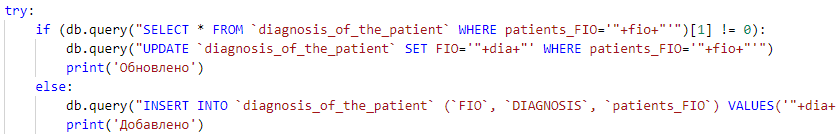
Примеры добавления данных:





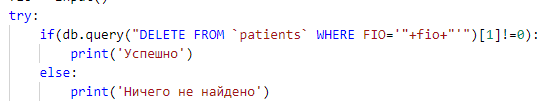
Пример изменения данных:

Изменение диагноза пациента в таблице “diagnosis\_of\_the\_patient”:



Пример удаления данных:

Удаление пациента из таблицы “patients”:



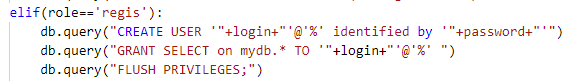
Пример поиска данных:

Поиск введенного заболевания в таблице “description\_disease”:



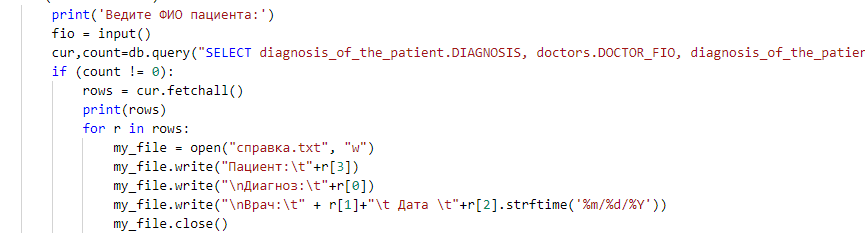
## **Доступ к данным**

Пример назначения прав доступа новому пользователю:

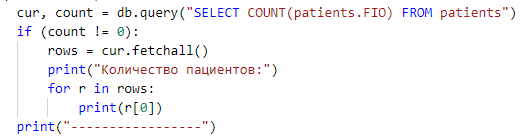


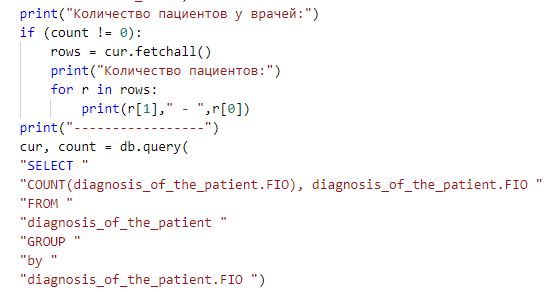
## **Предоставление запрашиваемых данных**

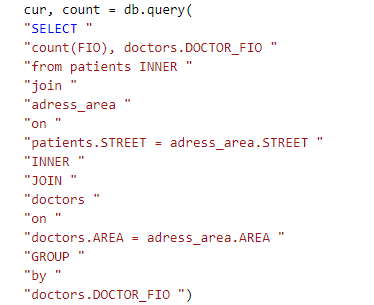
Пример генерации справки для пациента:

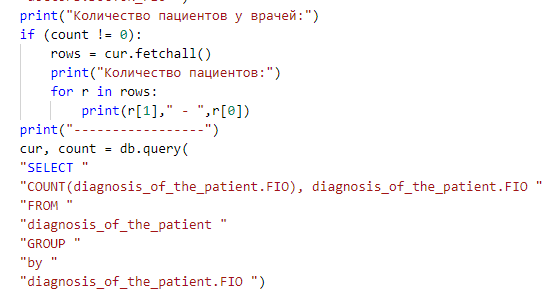


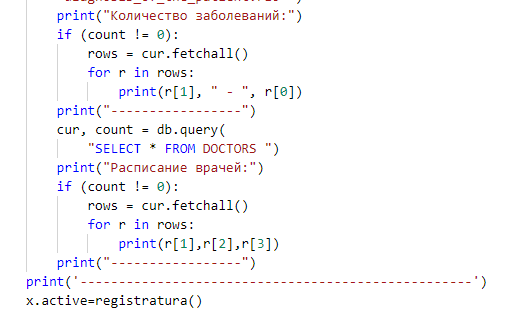
Пример генерации отчета о работе все поликлинике:









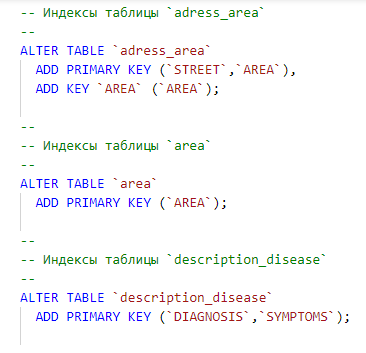


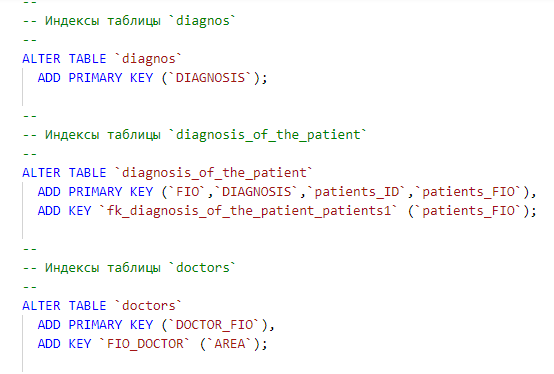
# **ОРГАНИЗАЦИЯ ЗАЩИТЫ БД**

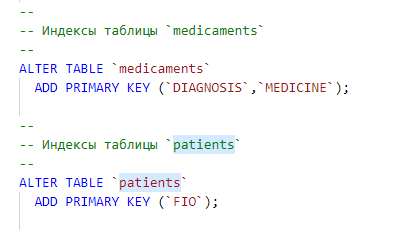
* 1. **Описание ограниченной целостности**

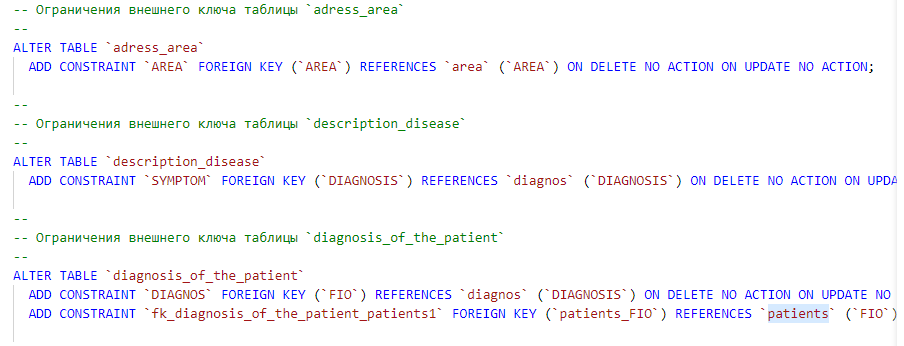
Из предыдущего пункта видно, что при создании информационных объектов, все переменные имеют значения NOT NULL.

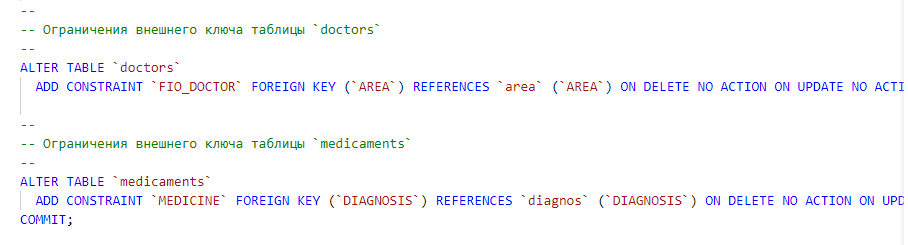
Далее были созданы уникальные первичные ключи, ограничения внешнего ключа и ограничения на обновление и удаление данных.

****

****

****

****

****

* 1. **Рекомендуемые средства физической защиты**

В качестве средств физической защиты данных чаще всего применяются резервное копирование и журналы транзакций.

Резервное копирование означает периодическое сохранение файлов базы данных на внешнем запоминающем устройстве. Оно выполняется тогда, когда состояние файлов базы данных является непротиворечивым. Резервная копия не должна создаваться и находиться на том же диске, на котором находится сама база данных, потому что при аварии диска базу невозможно будет восстановить. В случае сбоя базы данных восстанавливается на основе последней копии.

Рекомендуемые виды резервного копирования:

* Полная резервная копия - включает всю базу данных (все файлы БД, в том числе вспомогательные, состав которых зависит от СУБД).
* Частичная резервная копия включает часть БД, определённую пользователем.
* Инкрементная резервная копия - состоит только из тех блоков (страниц памяти), которые изменились со времени последнего резервного копирования. Создание же инкрементной копии происходит быстрее, чем полной, но оно возможно только после создания полной резервной копии.

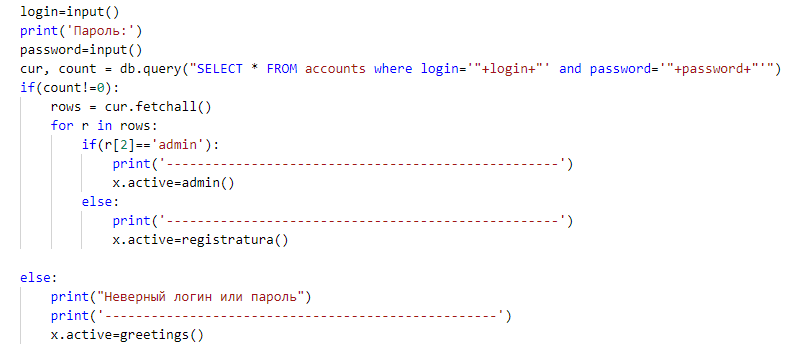
Как правило периодичность резервного копирования определяется администратором системы. Полное резервное копирование осуществляется раз в неделю (день, месяц), а частичное или инкрементное копирование раз в день (час, неделю).

В случае данной курсовой работы база данных будет храниться на другом сервере, мы копируем нашу базы данных на внешнее устройство и производим все необходимые процедуры для проведения резервного копирования.

* 1. **Процедура подтверждения личности**

Подтверждение подлинности личности заключается в доказательстве того, что пользователь является именно тем человеком, за которого себя выдаёт. Это выполняется путём парольной идентификации, которая заключается в присвоении каждому пользователю двух параметров: имени (login) и пароля (password). При входе в систему она запрашивает у пользователя его имя, а для подтверждения того, что это имя ввёл его владелец, система запрашивает пароль.

Реализация подтверждения личности пользователя в данной работе приведена ниже. При входе в приложение идет проверка введенного логина и пароля с логином и паролем из таблицы accounts.



# **ЗАКЛЮЧЕНИЕ**

Цель данной курсовой работы достигнута, все задачи выполнены. В итоге были произведены разработка и реализация реляционной базы данных для работников регистратуры поликлиники с помощью СУБД MySQL и языка Python.

В ходе выполнения работы были продемонстрированы основные моменты и нюансы создания и исполнения команд, отвечающие за взаимодействие базы данных с пользователем. Также произведено тестирование работы программы с последующим подробным и понятным описанием того, как нужно пользоваться данным приложением.

**список использованных источников**

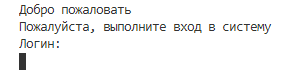
1. К. Дж. Дейт SQL и реляционная теория. Как грамотно писать код на SQL. – Пер. с англ. – СПб.: Символ-Плюс, 2010. – 480 с., ил.
2. Петкович Д. Microsoft SQL Server™ 2012. Руководство для начинающих: Пер. с англ. — СПб.: БХВ-Петербург, 2013. — 816 с.
3. Любанович Билл Простой Python. Современный стиль программирования. – СПб.: Питер, 2016. – 480 с.: – (Серия «Бестселлеры O’Reilly»).
4. Татарникова Т.М. Управление данными: Учебное пособие, СПБ: Изд. СПБГУТ, 2006.
5. Ризаев И.С., Яхина З.Т. Базы данных: Учебное пособие. Казань.: Изд-во Казан. гос. техн. ун-та. 2008. 240 с.
6. <https://docs.python.org/3/>
7. Как устроен Python. Гид для разработчиков, программистов и интересующихся. — СПб.: Питер, 2019. — 272 с.: ил. — (Серия «Библиотека программиста»).

**ПРИЛОЖЕНИЕ А. РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ БД**

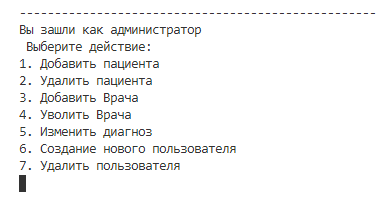
**Инструкция по получению доступа к содержимому БД**

Для того, чтобы пользователь смог воспользоваться всем функционалом БД, он должен запустить приложение. А затем ввести свой логин и пароль. Если пользователь не работал раннее с данным приложением, то он должен обратиться к администратору БД, чтобы тот его зарегистрировал и дал ему логин и пароль.

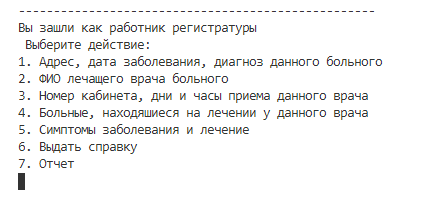
При входе в программу пользователю сначала необходимо ввести свой логин, а затем пароль:



Далее, если пользователь зашел как администратор, то перед ним появляется следующее меню:



А если пользователь зашел как работник регистратуры, то функционал будет следующий:

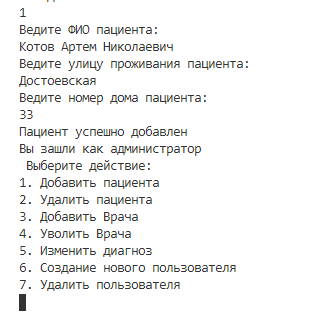


**Использование функций БД**

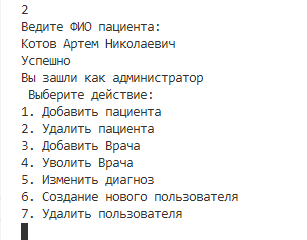
Для того, чтобы пользователь смог выполнить определенное действие, например, выдать справку, то он должен нажать соответствующую цифру.

Рассмотрим сначала весь функционал для администратора.

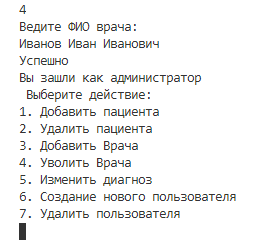
Для того, чтобы добавить нового пациента, администратору необходимо ввести цифру 1. Далее записать ФИО и адрес проживания пациента.



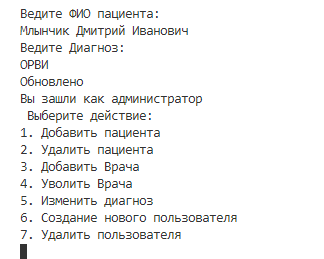
Для удаления пациента, администратору нужно ввести цифру 2 и ФИО пациента.



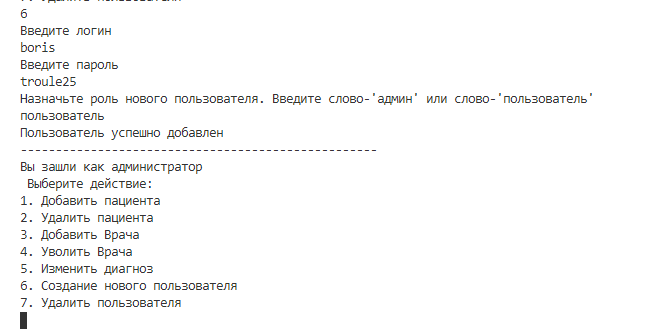
Для увольнения врача необходимо ввести цифру 4 и ФИО врача.



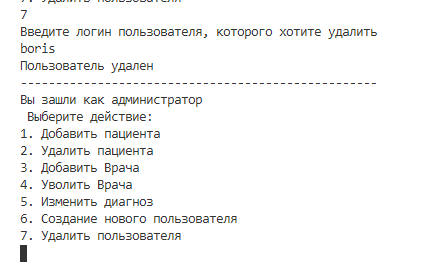
Введя цифру 5, пользователь может изменить диагноз пациента.



Нажимая на кнопку 6, администратор может создать нового пользователя и назначить ему определенные права:

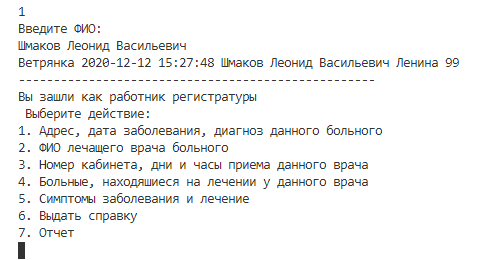


При вводе цифры 7, администратор также может и удалить пользователя, введя его логин:

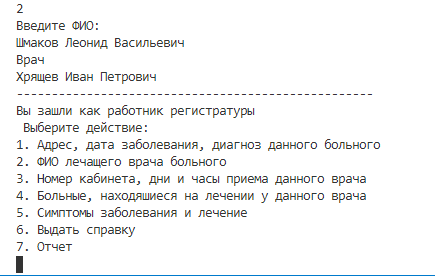


Теперь рассмотрим весь функционал для работника регистратуры поликлиники. Для того, чтобы пользователь выбрал определенное действие, он также должен нажать соответствующую цифру.

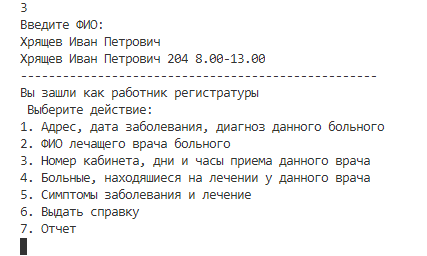
При нажатии на цифру 1, пользователь может получить информацию о болезни пациента, адресе его проживания и дате заболевания.



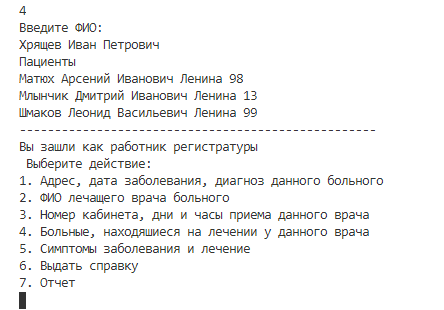
Вводя цифру 2 и ФИО больного, можно узнать его лечащего врача.



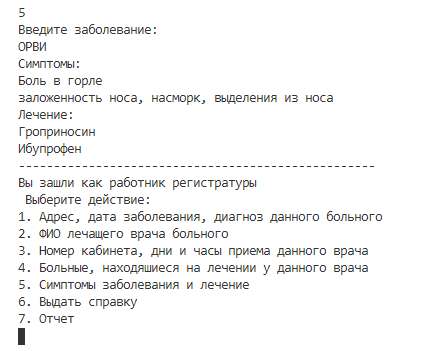
Для того, чтобы узнать номера кабинета врача и его рабочее время нужно ввести цифру 3, и затем ФИО врача.



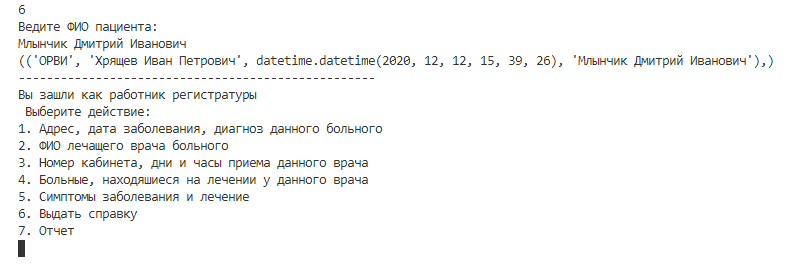
Нажимая на цифру 4 пользователю необходимо ввести ФИО врача, а затем на экран будет выдана информация о том, какие пациенты находятся у данного врача:



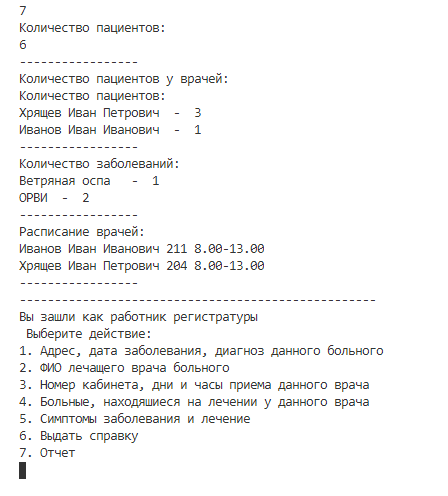
Работник регистратуры всегда может узнать симптомы заболевания и его лечение, предварительно нажав цифру 5 и записав название заболевания.



Для выдачи справки пациенту, пользователю необходимо ввести цифру 6. Далее ему также необходимо ввести ФИО пациента.



При нажатии кнопки 7 выдается отчет о работе всей поликлиники:

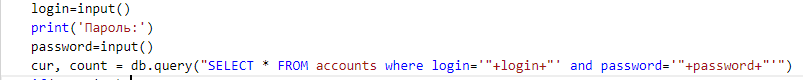


Для выхода из приложения администратору или работнику регистратуры необходимо нажать на кнопку ESC. Тогда пользователь попадет в самое начало программы, где ему необходимо ввести свой логин и пароль.

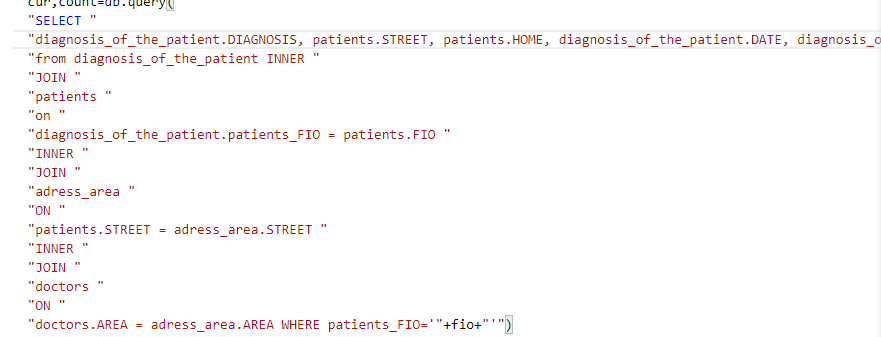
**ПРИЛОЖЕНИЕ Б. ЛИСТИНГ ПРОГРАММНОГО КОДА**

**Запросы, используемые в курсовой работе:**

Запрос на извлечение сведений о логине и пароле пользователя из таблицы accounts:



Запрос на извлечение сведений об адресе пациента, дате заболевания и его диагноза, содержащихся в таблицах diagnosis\_of\_the\_patient, patients, address\_area, doctors:



Запрос на извлечение сведений о ФИО лечащего врача больного из таблиц diagnosis\_of\_the\_patient, doctors:

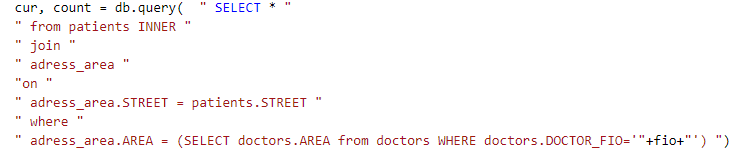




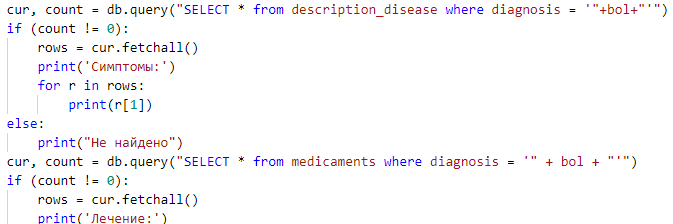
Запрос на извлечение сведений о ФИО лечащего врача больного из таблицы doctors:



Запрос на извлечение сведений о больных, находящихся на лечении у данного врача из таблиц patients, address\_area, doctors:



Запросы на извлечение сведений о симптомах заболевания и лечении из таблиц description\_disease и medicaments:



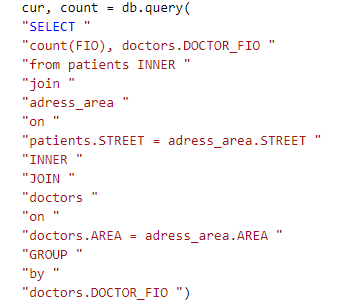
Запрос для выдачи справки:

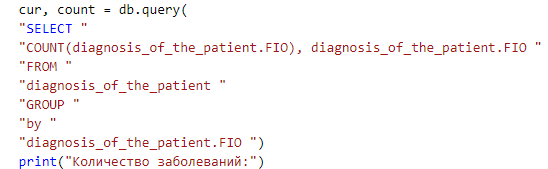




Запросы для выдачи отчета:



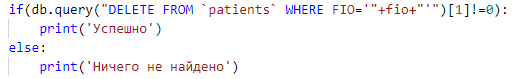




Запрос на добавление нового пациента в таблицу patients:



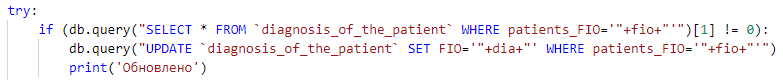
Запрос на удаление пациента из таблицы patients:



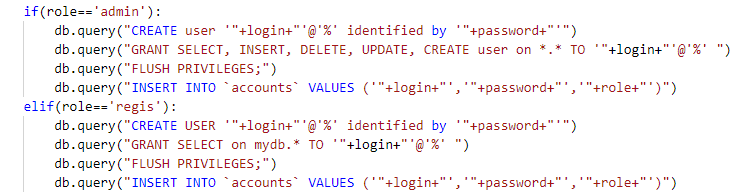
Запрос на удаление врача из таблицы DOCTORS:



Запрос на изменение диагноза пациента в таблице diagnosis\_of\_the\_patient:



Запрос для создания нового пользователя:



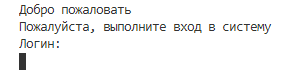
Запрос для удаления пользователя:



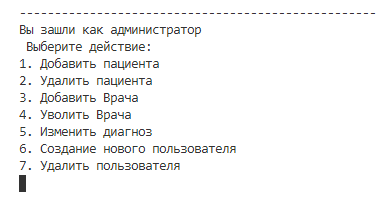
**Основные элементы интерфейса:**

Интерфейс данной курсовой работы был разработан в консоли.

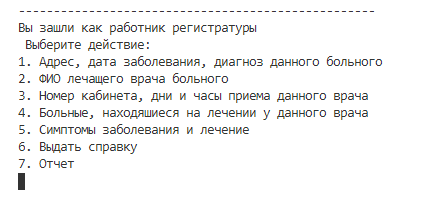
Элементы интерфейса для ввода пользователем логина, а затем и пароля:



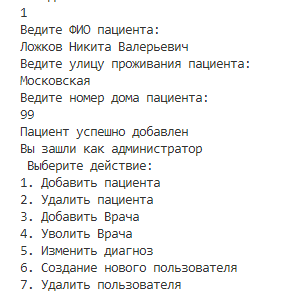
Если пользователь зашел как администратор, то перед ним появляется меню:



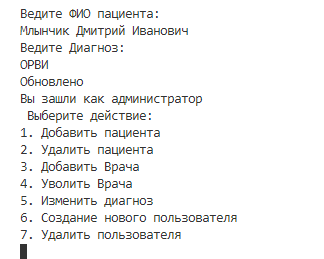
А если он зашел как работник регистратуры, то функционал и интерфейс меню будет следующий:



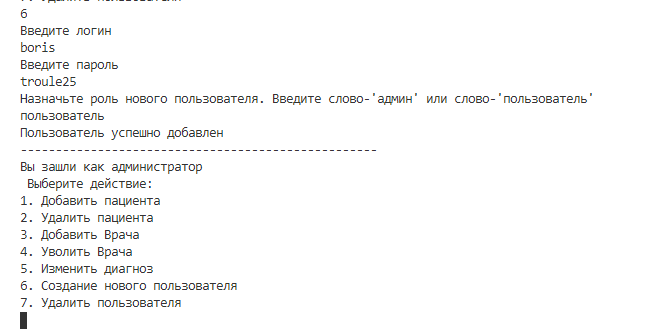
Элементы взаимодействия пользователя с программой и добавление нового пациента.

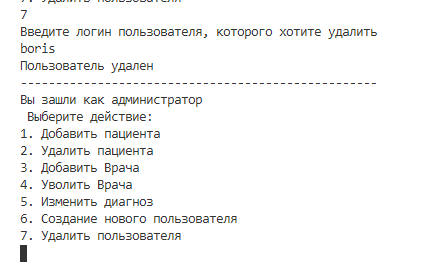


Изменение диагноза пациента.

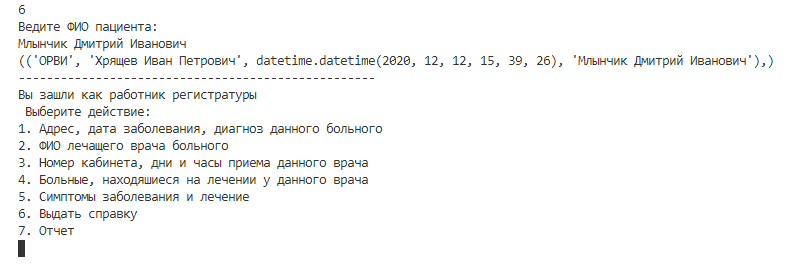


Интерфейс создания и удаления пользователя





Выдача справки для больного



Вывод отчета о работе всей поликлинике

