

Федеральное государственное бюджетное образовательное
учреждение высшего образования
**РЫБИНСКИЙ ГОСУДАРСТВЕННЫЙ АВИАЦИОННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ П.А. СОЛОВЬЕВА**

Факультет радиоэлектроники и информатики
Кафедра математического и программного обеспечения электронных
вычислительных средств

КУРСОВОЙ ПРОЕКТ

по дисциплине
«БАЗЫ ДАННЫХ»
на тему:
«Диспансерный учёт больных»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Студенты группы ИПБ-18 _____ Зернов И.В., Кондратенко М.М.
Руководители _____ Шаров В.Г., Задорина Н.А.

Рыбинск 2021

Оглавление

1 Постановка задачи.....	4
2 Введение.....	5
2.1 Актуальность разработки.....	5
2.2 Цель работы.....	5
3 Описание предметной области.....	6
3.1 Общее описание предметной области.....	6
3.2 Описание входных документов.....	6
3.3 Описание выходных документов.....	7
Инфологическая модель.....	8
3.1. Сущности и их первичные ключи.....	8
3.2. ER-диаграмма.....	8
3.3. Бизнес-правила.....	8
Даталогическая модель.....	10
4.1. Предварительные отношения:.....	10
4.2. Перечень не ключевых атрибутов:.....	10
4.3. Полный набор отношений с распределенными не ключевыми атрибутами:.....	11
4.4. Набор функциональных зависимостей для каждого отношения, подтверждающий форму НФБК.....	11
4.5. Сводная таблица отношений.....	12
4.6. Структура отношений базы данных.....	12
4.7. Структура базы данных.....	14
5 Реализация серверной части.....	15

6 Реализация приложения	15
6.1. Модель функционирования системы	15
6.2. Схема меню приложения.....	15
5.3. Окна приложения	17
Заключение	33
Список литературы	34
Приложение	35

1 Постановка задачи

Поликлиника ведет учет больных, имеющих хронические заболевания. В зависимости от заболевания больной должен осматриваться врачами нескольких специальностей. У каждого больного имеется один ведущий врач. Каждому больному определяется дата очередного профилактического осмотра.

Система должна обеспечивать:

- прием на работу врача-специалиста;
- увольнение врача;
- перепрофилирование врача;
- регистрацию нового больного;
- запись больного к ведущему врачу с выдачей направления к соответствующим специалистам и назначение даты нового профосмотра;
- получение справок о работающих врачах, о больных с некоторым заболеванием, о заболеваниях диспансерных больных, о перечне специалистов, которые должны курировать данное заболевание, список больных некоторого врача, о больных, подлежащих осмотру в некоторый день и др.

2 Введение

2.1 Актуальность разработки

При отсутствии электронного документооборота в диспансере необходимо организовать сложный процесс, включающий в себя ведение и организацию карточек больных, информацию о врачах и их специальностях, а также систему записи пациентов на приём.

Создание приложения позволит не только уменьшить количество ошибок, связанных с человеческим фактором, но и за счёт связи всех компонент системы (информация о врачах, больных, болезнях и т.д.) ускорить и упростить работу персонала с документами.

2.2 Цель работы

Цель создания программы – упростить процедуру управления документооборотом диспансера.

3 Описание предметной области

3.1 Общее описание предметной области

Управление документами в диспансере – это сложный и трудозатратный процесс, который требует повышенной внимательности от сотрудника. Администратору необходимо вести учёт карточек пациентов, реестр приёмов в соответствии с расписанием врачей и их специальностью, отслеживать увольнение и прием на работу врачей, выдавать различные справки и др. Зачастую, все эти данные никак не связаны между собой и находятся в разных местах.

При приёме врача на работу администратору необходимо внести карточку врача в реестр. В карточку вносятся ФИО врача, его специальность и кабинет, где будет происходить приём пациентов.

Запись пациентов на приём обязательно вносится в реестр администратором. При внесении записи в реестр пациенту выдается талон с ФИО врача, ФИО самого пациента и с назначенной датой. Так как запись внесена в реестр, врач видит список пациентов, которых он курирует и которым назначена запись на определённую дату.

Создание приложения позволит автоматизировать процессы, описанные выше и связать данные, тем самым исключая ошибки, связанные с вниманием персонала.

2.2 Описание входных документов

Для правильного функционирования документооборота диспансера необходимы следующие документы:

- заявка на обслуживание пациента в диспансере (ФИО, дата рождения, прописка, СНИЛС, заболевание);
- заявление о приёме на работу врача (ФИО, специальность);
- заявление на увольнение врача из диспансера (ФИО);

- форма для перевода пациента к другому врачу (ФИО, ФИО нового врача);

2.3 Описание выходных документов

В качестве выходных документов система предоставляет:

- список врачей с указанием специальности и заболеваний, относящихся к этой специальности,
- талон о записи на приём с указанием даты и ФИО врача,
- информацию о пациенте (выдается только самому пациенту),
- информацию о пациентах для врачей.

3 Инфологическая модель

3.1 Сущности и их первичные ключи

- Пациент (НомПац)
- Заболевание (НомЗаб)
- Врач (НомВрач)
- ВрачСпец (НомСпец)

3.2 ER-диаграмма

ER-диаграмма представлена на рисунке 1.

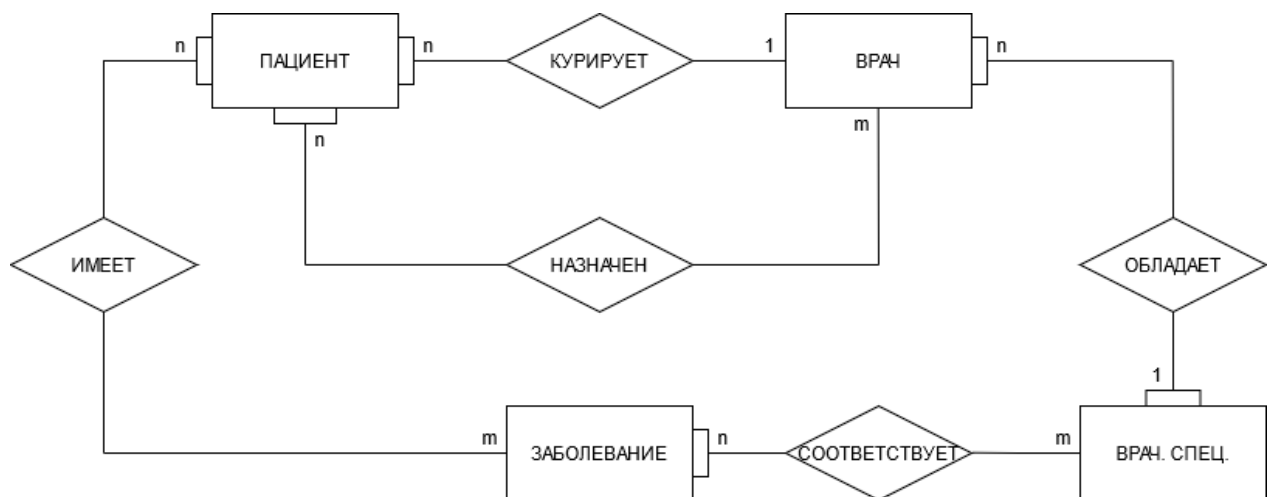


Рис. 1 - ER-диаграмма, описывающая базу данных приложения.

3.3 Бизнес-правила

Пациент - Врач (курирует)

1. Пациент курируется только одним врачом
2. Врач может курировать несколько пациентов
3. Пациента обязательно курирует хотя бы один врач
4. Врач может не курировать пациента

Пациент - Врач (назначен)

5. Пациент может быть назначен на осмотр к нескольким врачам
6. Врач может осматривать несколько пациентов
7. Пациент обязательно назначен на осмотр ко врачу

8. Врач не обязательно осматривает пациентов

Пациент – Заболевание

9. Несколько пациентов могут иметь одно заболевание

10. Один пациент может иметь несколько заболеваний

11. Пациент имеет хотя бы одно заболевание

12. Может существовать заболевание, не имеющееся ни у одного пациента

Врач – Врачебная Специальность

13. Врач имеет только одну врачебную специальность

14. Несколько врачей могут обладать одной врачебной специальностью

15. Врач обязательно обладает врачебной специальностью

16. Врачебная специальность имеется хотя бы у одного врача

Заболевание – Врачебная Специальность

17. Одному заболеванию может соответствовать несколько врачебных специальностей

18. Врачебная специальность может соответствовать нескольким заболеваниям

19. Заболевание соответствует хотя бы одной врачебной специальности

20. Врачебной специальности не обязательно соответствует заболевание

4 Даталогическая модель

4.1 Предварительные отношения:

- Пациент (НомПац, ..., НомВрач)
- Заболевание (НомЗаб, ...)
- Врач (НомВрач, ..., НомСпец)
- ВрачСпец (НомСпец, ...)
- ЗабПац (НомПац, НомЗаб, ...)
- Назначение (НомВрач, НомПац, ...)
- ЗабВрачСпец (НомЗаб, НомСпец, ...)

4.2 Перечень не ключевых атрибутов:

1. Пациент:
 - ФИО
 - Дата Рождения
 - Прописка
 - СНИЛС
2. Заболевание:
 - Название
3. Врач:
 - ФИО
 - Кабинет
4. ВрачСпец
 - Название
5. ЗабПац:
 - ДатаЗаболевания
6. Назначение
 - Дата
7. ЗабВрачСпец:

- ДатаПоследнейКвалификации

4.3 Полный набор отношений с распределенными не ключевыми атрибутами:

- Пациент (НомПац, ФИО, ДатаРождения, Прописка, СНИЛС, НомВрач)
- Заболевание (НомЗаб, Название)
- Врач (НомВрач, ФИО, Кабинет, НомСпец)
- ВрачСпец (НомСпец, Название)
- ЗабПац (НомПац, НомЗаб, ДатаЗаболевания)
- Назначение (НомВрач, НомПац, Дата)
- ЗабВрачСпец (НомЗаб, НомСпец, ДатаПоследнейКвалификации)

4.4 Набор функциональных зависимостей для каждого отношения, подтверждающий форму НФБК

1. Функциональные зависимости отношения Пациент:
 - НомПац → НомВрач
 - НомПац → ФИО
 - НомПац → Дата Рождения
 - НомПац → Прописка
 - НомПац → СНИЛС
2. Функциональные зависимости отношения Заболевание:
 - НомЗаб → Название
3. Функциональные зависимости отношения Врач:
 - НомВрач → НомСпец
 - НомВрач → ФИО
 - НомВрач → Кабинет
4. Функциональные зависимости отношения ВрачСпец:
 - НомСпец → Название
5. Функциональные отношения ЗабПац:

- ЗабПац → ДатаЗаболевания
- ЗабПац → Назначение
- ЗабПац → Дата

6. Функциональные зависимости отношения ЗабВраСпец:

- ЗабВрачСпец → ДатаПоследнейКвалификации

4.5 Сводная таблица отношений

Имя таблицы	Описание
Пациент	Таблица пациентов
Врач	Таблица врачей
Заболевание	Таблица заболеваний
Назначение	Таблица назначений
ВрачСпец	Таблица связи врача и специальности
ЗабПац	Таблица связи заболевания и пациента
ЗабВрачСпец	Таблица связи заболевания и специальности

4.6 Структура отношений базы данных

Наименование информационного объекта	Наименование атрибута	Наименование в БД	Тип	Длина	Признак ключа
Пациент	НомПац	idPatient	INT	-	пк
	ФИО	fio	VARCHAR	45	
	Адрес	address	VARCHAR	45	
	СНИЛС	SNILS	VARCHAR	56	
	Заболевание	Illness_idIllness	INT	-	
		Doctor_idDoctor	INT	-	вк
		Speciality_idSpeciality	INT	-	вк
	Дата рождения	birthDate	DATE	-	
Врач	НомВрач	idDoctor	INT	-	пк
	ФИО	FIO	VARCHAR	45	

	Кабинет	roomNumber	INT	-	
	НомСпец	Specialty	INT	-	ВК
Заболевание	НомЗаб	idIllness	INT	-	ПК
	Название	name	VARCHAR	45	
Назначение	НомНаз	idAppointment	INT	-	ПК
	НомВрач	Doctor_idDoctor	INT	-	ВК
	НомПац	Patient_idPatient	INT	-	ВК
	Дата	date	DATE	-	
ВрачСпец	НомСпец	idSpeciality	INT	-	ПК
	Название	name	VARCHAR	45	
ЗабПац	НомПац	Patient_idPatient	INT	-	ВК
	НомЗаб	Illness_idIllness	INT	-	ВК
ЗабВрачСпец	НомЗаб	Illness_idIllness	INT	-	ВК
	НомСпец	Speciality_idSpeciality	INT	-	ВК

4.7 Структура базы данных

Структура базы данных представлена на рисунке 2.

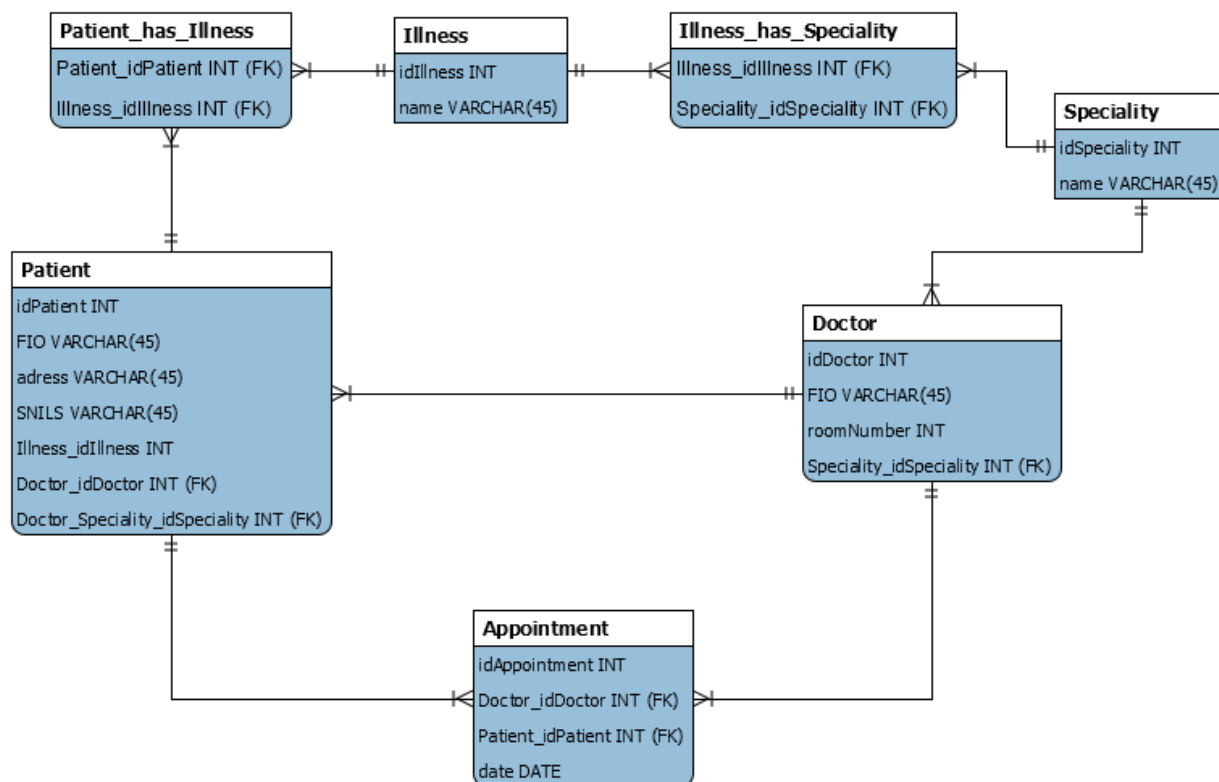


Рис. 2 - Структура базы данных.

5 Реализация серверной части

Работа с базой данных осуществляется с помощью библиотеки MySQL Database Connector для Django.

Сначала устанавливается соединение с базой данных с использованием логина и пароля пользователя, созданного специально для работы только с базой данных системы. После этого с использованием специальных функций библиотеки MySQL Database Connector выполняются SQL запросы к базе данных для получения необходимой информации. Примеры запросов приведены в Приложении к курсовой работе (см. Листинг 1, Листинг 2). После получения данных соединение с БД закрывается.

СУБД не использует ни триггеры, ни хранимые процедуры. Все запросы формируются динамически в приложении.

6 Реализация приложения

6.1 Модель функционирования системы

Схема взаимодействия технических средств представлена на рисунке 3.



Рис 3 - Схема взаимодействия технических средств.

6.2 Схема меню приложения

Схема меню приложения представлена на рисунке 4.

Строка меню находится в левой части экрана и содержит название программы, а также кнопки меню, в котором доступны все разделы программы.

Рабочая область занимает основную часть экрана. В рабочей области главного экрана программы находятся элементы, позволяющие реализовывать заданные функции:

- прием на работу врача-специалиста;
- увольнение врача;
- перепрофилирование врача;
- регистрацию нового больного;
- запись больного к ведущему врачу с выдачей направления к соответствующим специалистам и назначение даты нового посещения;
- получение справок о работающих врачах;
- получение справок о больных с некоторым заболеванием;
- получение справок о заболеваниях диспансерных больных;
- получение справок о перечне специалистов, которые должны курировать данное заболевание;
- получение список больных некоторого врача.

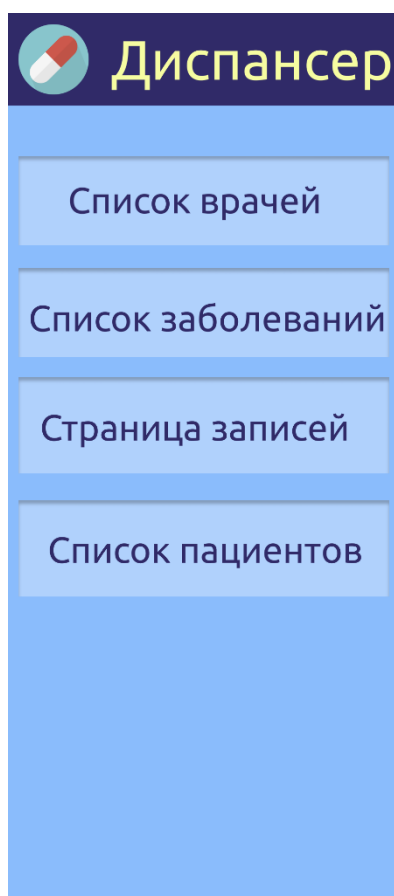


Рис. 4 – Схема меню приложения.

6.3 Окна приложения

При запуске программы, перед пользователем появится окно с информацией о диспансере (Рисунок 5). У пользователя есть возможность выбрать в левой части экрана один из пунктов меню, чтобы выполнить желаемую задачу.

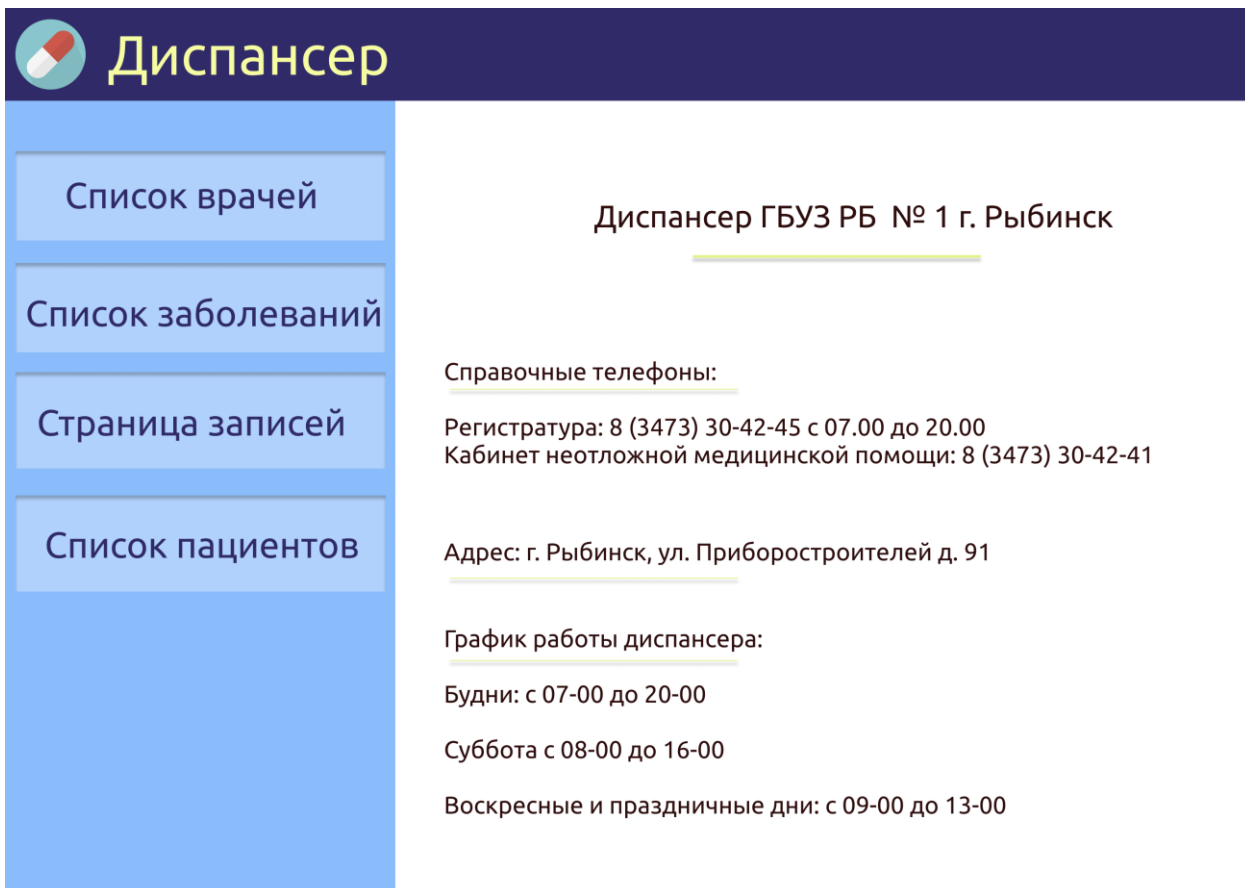


Рис. 5 – Главное окно приложения.

При нажатии на кнопку «Список врачей» отобразится соответствующее окно (Рисунок 6). Пользователь видит полный список врачей с их специальностью.

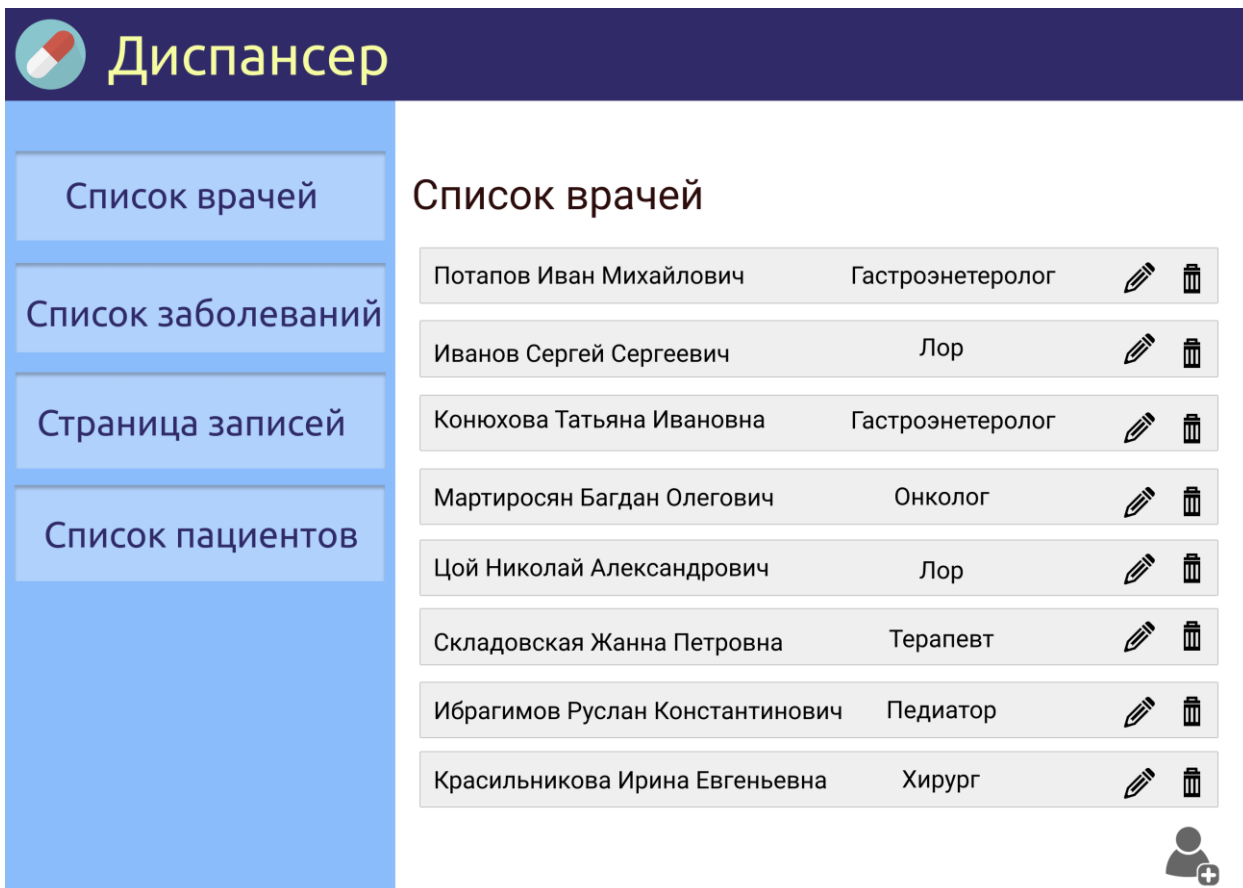



Рис. 6 – Окно списка врачей.

При нажатии на кнопку «Редактировать» отобразится соответствующее окно (Рисунок 7). В данном окне можно изменить ФИО, номер кабинета, специальность. При нажатии на кнопку «Сохранить» на странице редактирования изменения сохранятся и в БД. При нажатии кнопки «Удалить» запись данного врача удаляется из БД и удаляется из списка врачей. При нажатии на кнопку «Добавить» отобразится окно (Рисунок 8) с полями ФИО, номер кабинета и специальность. После их заполнения и нажатия кнопки «Сохранить» все изменения вносятся в БД. При нажатии на ФИО отобразится окно (Рисунок 9), где пользователь видит все данные о враче: ФИО, номер кабинета, специальность, кого курирует врач.

 **Диспансер**

Список врачей

Список заболеваний

Страница записей

Список пациентов

Страница редактирования врача

ФИО:

Красильникова Ирина Евгеньевна

Номер кабинета:


266

Специальность:

Терапевт

Сохранить

Рис. 7 – Окно редактирования врача информации о враче.

 **Диспансер**

Список врачей

Список заболеваний

Страница записей

Список пациентов

Страница добавления врача

ФИО:

Красильникова Ирина Евгеньевна

Номер кабинета:

266

Специальность:

Терапевт

Сохранить

Рис. 8 – Окно добавления врача.



Диспансер

Список врачей

Список заболеваний

Страница записей

Список пациентов

Детальная страница врача

ФИО:

Красильникова Ирина Евгеньевна

Номер кабинета:

266

Специальность:

Терапевт

Кого курирует:

Маргиева Юлия Александровна

Рис. 9 – Окно с информацией о враче.

При нажатии на кнопку «Список заболеваний» отобразится соответствующее окно (Рисунок 10). Пользователь видит перечень заболеваний, зарегистрированных в диспансере. При нажатии кнопки «Добавить» пользователю отобразится окно «Страница добавления заболевания» (Рисунок 11), где можно внести новое заболевание. После нажатия кнопки «Сохранить» новое заболевание вносится в БД и отображается в списке заболеваний.

[Список врачей](#)[Список заболеваний](#)[Страница записей](#)[Список пациентов](#)

Список заболеваний

Бронхит

Гайморит

Ангина

Туберкулёз

Воспаление лёгких


Герпес

Гастрит

Менингит



Рис. 10 – Окно со список заболеваний.

Диспансер

Список врачей

Список заболеваний

Страница записей

Список пациентов

Страница добавления заболевания

Название:

Бронхит

Сохранить

Рис. 11 – Окно добавления заболевания.

При нажатии на название заболеваний отобразится соответствующее окно (Рисунок 12). Пользователь видит название заболевания, список врачей, которые могут лечить данное заболевание, и список больных с данным заболеванием.

заболеванием.

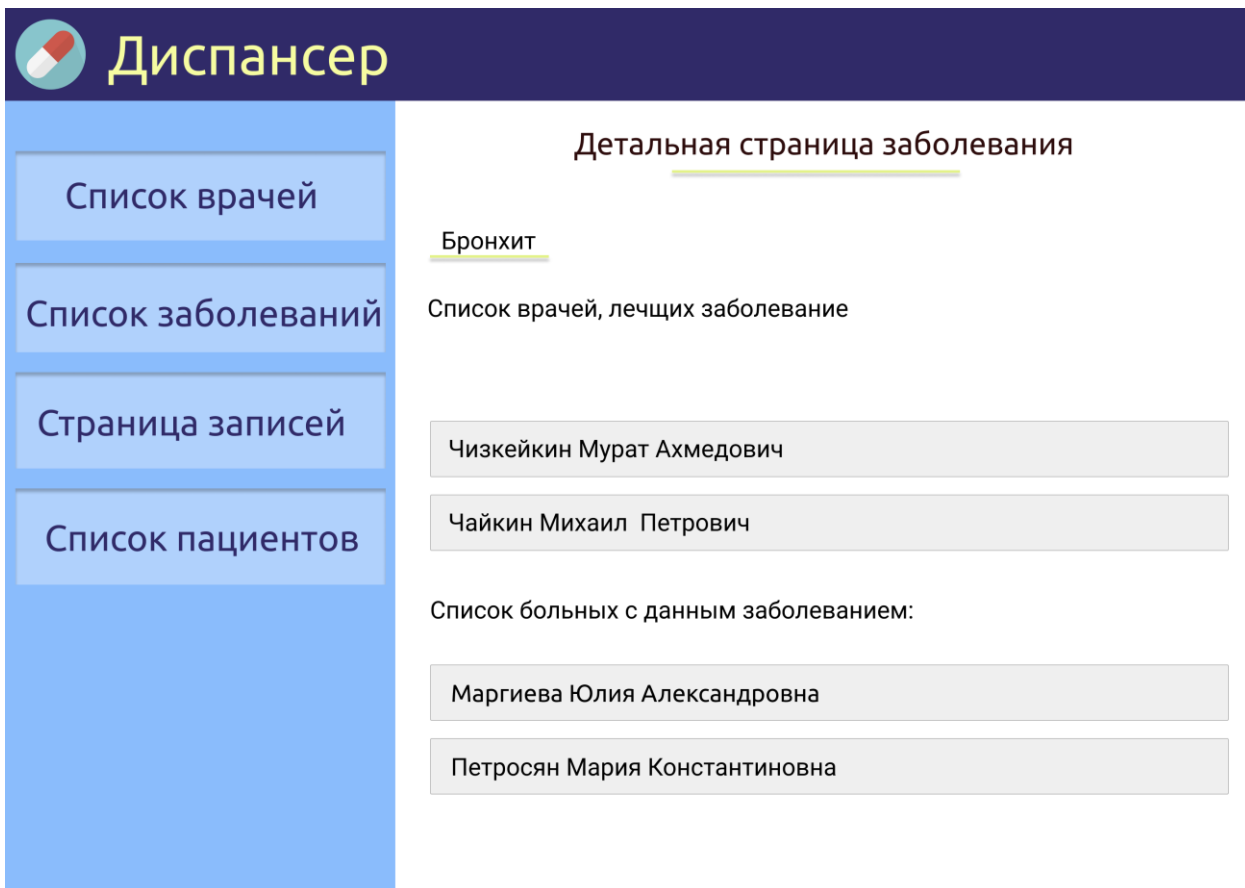


Рис. 12 – Окно с детальной информацией о заболевании.

При нажатии на кнопку «Страница записей» отобразится соответствующее окно (Рисунок 13).

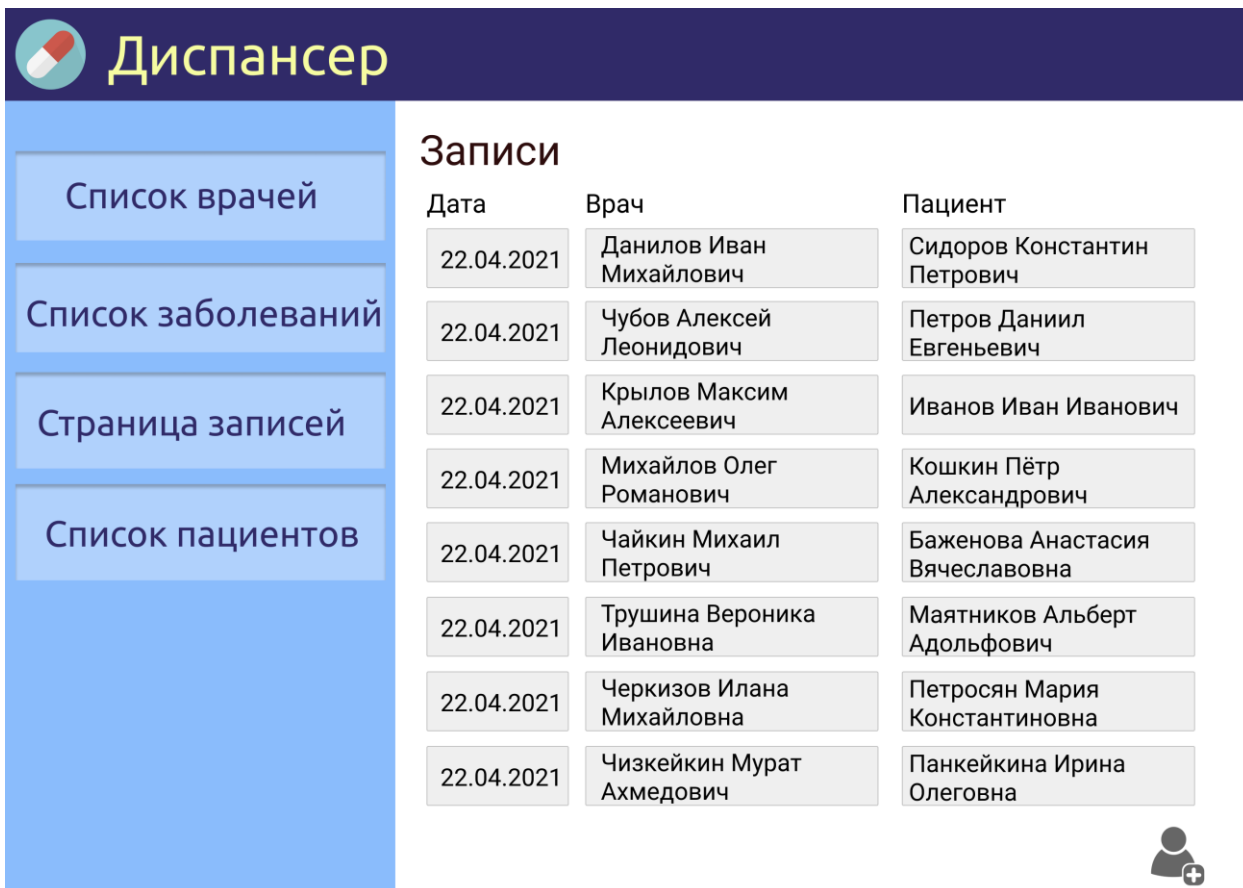


Рис. 13 – Окно с записями на приём.

При нажатии на кнопку «Добавить» отобразиться окно (Рисунок 17) «Добавление записи», где можно выбрать врача, пациента и дату. После нажатии кнопки «Записать» отображается окно «Талон» (Рисунок 19).

При нажатии на кнопку «Список пациентов» отобразится соответствующее окно (Рисунок 14). Выводится список всех пациентов в диспансере. При нажатии на ФИО пациента отображается «Детальная страница пациента» (Рисунок 15), где есть поля ФИО, день рождения, адрес, СНИЛС, доктор, который курирует пациента, дата посещения. При нажатии на кнопку «Добавить» отобразится окно «Страница добавления пациента» (Рисунок 16), где заполняются поля ФИО, день рождения, СНИЛС, курирующий врач и болезни. После нажатия кнопки «Сохранить» все данные вносятся в БД.

Список врачей

Список заболеваний

Страница записей

Список пациентов

Список пациентов

Маргиев Юлия Александровна

Сидоров Константин Петрович

Иванов Иван Иванович

Беляева Милана Сергеевна

Панкейкина Ирина Олеговна

Кошкин Пётр Александрович

Петросян Мария Константиновна

Маятников Альберт Адольфович



Рис. 14 – Окно со список всех пациентов.

Список врачей

Список заболеваний

Страница записей

Список пациентов

Детальная страница пациента

ФИО:



Маргиева Юлия Александровна

День рождения

10.12.1997

СНИЛС:

142-511-371-11

Курирующий врач:

Красильникова Ирина Евгеньевна

Дата посещения:

27.07.2021

Рис. 15 – Окно детальной страницы пациента.

 **Диспансер**

Список врачей

Список заболеваний

Страница записей

Список пациентов

Страница добавление пациента

ФИО:

Маргиева Юлия Александровна

День рождения

10.12.1997

СНИЛС:

142-511-371-11

Курирующий врач:

Красильникова Ирина Евгеньевна


Болезни:

Бронхит, ангина

Сохранить

Рис. 16 – Окно детальной страницы пациента.

При нажатии на кнопку «Запись» на «Детальной странице пациента» отобразится соответствующее окно (Рисунок 17). Отобразится окно «Записи» с полями пациент, доктор, дата, где пользователь может записать пациента к доктору на нужную дату. При нажатии на кнопку «Запись» вся информация вносится в БД. При нажатии на кнопку «Редактировать» на «Детальной странице пациента» отобразиться страница (Рисунок 18) редактирования пациента с полями ФИО, адрес, СНИЛС, доктор, дата. Пользователь может менять данные поля.

 **Диспансер**

Список врачей

Список заболеваний

Страница записей

Список пациентов

Запись

Маргиева Юлия Александровна

Красильникова Ирина Евгеньевна

27.07.2021

Записать

Рис. 17 – Окно записи пациента на осмотр.

 **Диспансер**

Список врачей

Список заболеваний

Страница записей

Список пациентов

Страница редактирования пациента

ФИО:

Маргиева Юлия Александровна

День рождения

10.12.1997

СНИЛС:

142-511-371-11

Курирующий врач:

Красильникова Ирина Евгеньевна

Дата посещения:

27.07.2021

Сохранить

Рис. 18 – Окно страницы редактирования пациента.

При нажатии на кнопку «Записать» на странице «Запись» пользователю отобразится «Талон», где будет информация о ФИО пациента, к какому врачу он записан и на какую дату (Рисунок 19).

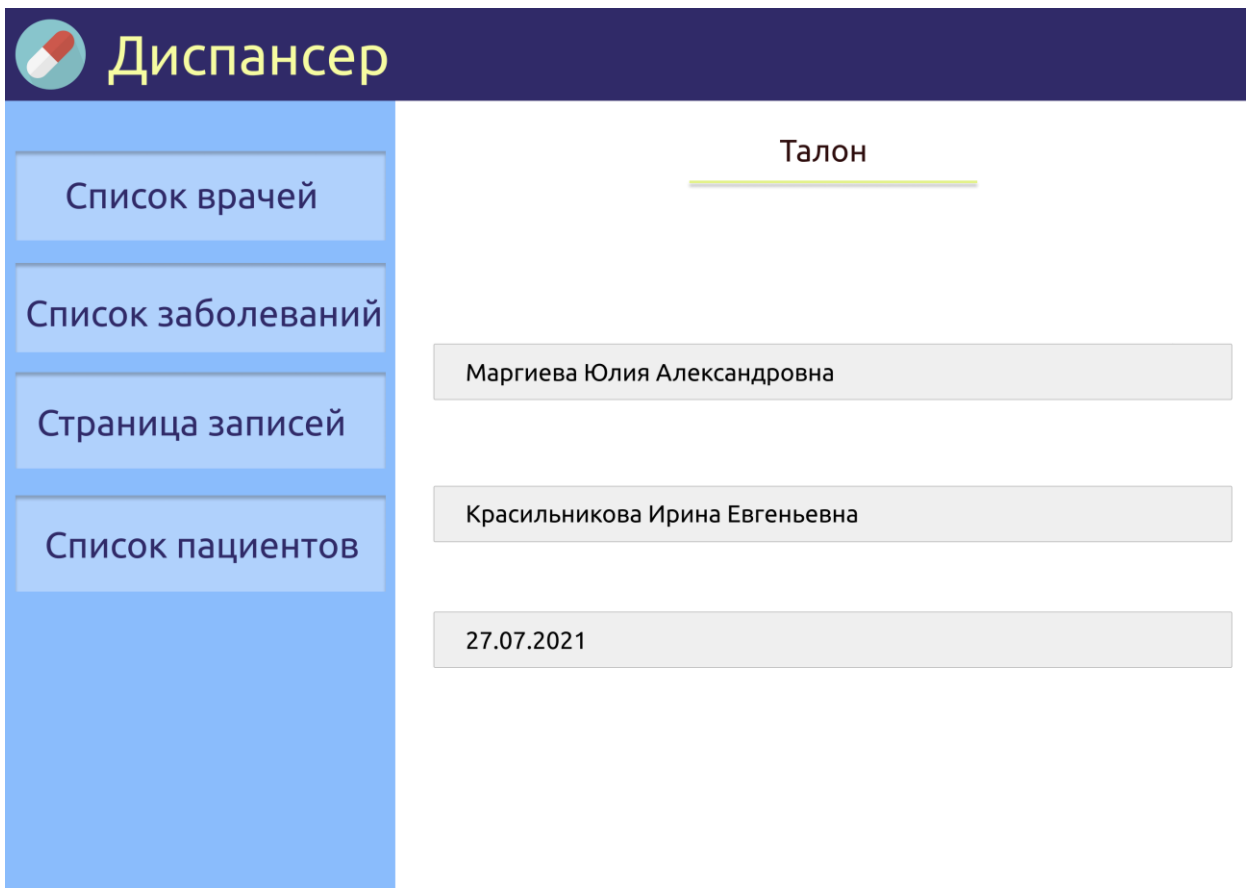


Рис. 19 – Окно со страницей «Талон».

Заключение

В результате проделанной работы была спроектирована реляционная база данных на основе построения ER–диаграммы сущность-связь, составлены предварительные отношения по спроектированной диаграмме.

В процессе создания программы были получены навыки и новые знания в работе с фреймворком Nuxt, работающем с языком высокого уровня JavaScript. А также с фреймворком Django, работающем с языком высокого уровня Python. Улучшены навыки работы с декларативным языком программирования SQL, а также был освоен движок для кроссплатформенных баз данных MySQL Database Connector.

В процессе работы было создано веб приложение для учёта диспансерных больных, обеспечивающее ускорение и надежность ведения документооборота.

Список литературы

1. Лутц, М. Программирование на Python / М. Лутц. - 4-е изд., - СПб: Символ-Плюс, 2011.
2. Лазаро Исси Коэн / Джозеф Исси Коэн Полный справочник по HTML, CSS и JavaScript. -М.: ЭКОМ Паблишерз, 2016. - 311 с.
3. Документация Nuxt: сайт. – URL: <https://ru.nuxtjs.org/docs/2.x/get-started>
(дата обращения: 11.04.2021 - 20.06.2021)
4. Документация Django: сайт. – URL:
<https://docs.djangoproject.com/en/3.2/>
(дата обращения: 11.04.2021 - 20.06.2021)

Приложение

Вывод талонов в excel

Таблица 1

Врач	Пациент	Дата
Красильникова Ирина Евгеньевна	Маргиева Юлия Александровна	27.07.2021
Данилов Иван Михайлович	Сидоров Константин Петрович	01.07.2021
Крылов Максим Алексеевич	Иванов Иван Иванович	01.07.2021
Трушина Вероника Ивановна	Маятников Альберт Адольфович	02.07.2021
Черкизова Илана Михайловна	Петросян Мария Константиновна	04.07.2021

Построение диаграммы заполненности диспансера на основе записей на определённую дату



Рис. 20 – Заполненность диспансера.

Построение диаграммы загрузки врачей



Рис. 21 –Загруженность врачей.

Листинг 1 создание таблиц в БД

Создание таблицы «Заболевание»:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Illness` (  
  `idIllness` INT NOT NULL,  
  `name` VARCHAR(45) NULL,  
  PRIMARY KEY (`idIllness`))  
ENGINE = InnoDB;
```

Создание таблицы «Специальность»:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Speciality` (  
  `idSpeciality` INT NOT NULL,  
  `name` VARCHAR(45) NULL,  
  PRIMARY KEY (`idSpeciality`))  
ENGINE = InnoDB;
```

Создание таблицы «Доктор»:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Doctor` (  
  `idDoctor` INT NOT NULL,  
  `FIO` VARCHAR(45) NULL,  
  `roomNumber` INT NULL,
```

```

`Speciality_idSpeciality` INT NOT NULL,
PRIMARY KEY (`idDoctor`, `Speciality_idSpeciality`),
INDEX `fk_Doctor_Speciality_idx` (`Speciality_idSpeciality`
ASC) VISIBLE,
CONSTRAINT `fk_Doctor_Speciality`
FOREIGN KEY (`Speciality_idSpeciality`)
REFERENCES `mydb`.`Speciality` (`idSpeciality`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

Создание таблицы «Пациент»:

```

CREATE TABLE IF NOT EXISTS `mydb`.`Patient` (
`idPatient` INT NOT NULL,
`FIO` DATE NULL,
`adress` VARCHAR(45) NULL,
`SNILS` VARCHAR(45) NULL,
`Illness_idIllness` INT NOT NULL,
`Doctor_idDoctor` INT NOT NULL,
`Doctor_Speciality_idSpeciality` INT NOT NULL,
PRIMARY KEY (`idPatient`, `Illness_idIllness`,
`Doctor_idDoctor`, `Doctor_Speciality_idSpeciality`),
INDEX `fk_Patient_Doctor1_idx` (`Doctor_idDoctor` ASC,
`Doctor_Speciality_idSpeciality` ASC) VISIBLE,
CONSTRAINT `fk_Patient_Doctor1`
FOREIGN KEY (`Doctor_idDoctor` ,
`Doctor_Speciality_idSpeciality`)
REFERENCES `mydb`.`Doctor` (`idDoctor` ,
`Speciality_idSpeciality`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

Создание таблицы «Запись»:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Appointment` (  
  `idAppointment` INT NOT NULL AUTO_INCREMENT,  
  `Doctor_idDoctor` INT NOT NULL,  
  `Patient_idPatient` INT NOT NULL,  
  `date` DATE NULL,  
  PRIMARY KEY (`idAppointment`, `Doctor_idDoctor`,  
  `Patient_idPatient`),  
  INDEX `fk_Appointment_Doctor1_idx` (`Doctor_idDoctor` ASC)  
  VISIBLE,  
  INDEX `fk_Appointment_Patient1_idx` (`Patient_idPatient` ASC)  
  VISIBLE,  
  CONSTRAINT `fk_Appointment_Doctor1`  
    FOREIGN KEY (`Doctor_idDoctor`)  
    REFERENCES `mydb`.`Doctor` (`idDoctor`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION,  
  CONSTRAINT `fk_Appointment_Patient1`  
    FOREIGN KEY (`Patient_idPatient`)  
    REFERENCES `mydb`.`Patient` (`idPatient`)  
    ON DELETE NO ACTION  
    ON UPDATE NO ACTION)  
ENGINE = InnoDB;
```

Создание таблицы связей таблиц «Заболевание» и «Специальность»:

```
CREATE TABLE IF NOT EXISTS `mydb`.`Illness_has_Speciality` (  
  `Illness_idIllness` INT NOT NULL,  
  `Speciality_idSpeciality` INT NOT NULL,  
  PRIMARY KEY (`Illness_idIllness`, `Speciality_idSpeciality`),  
  INDEX `fk_Illness_has_Speciality_Speciality1_idx`  
  (`Speciality_idSpeciality` ASC) VISIBLE,  
  INDEX `fk_Illness_has_Speciality_Illness1_idx`  
  (`Illness_idIllness` ASC) VISIBLE,  
  CONSTRAINT `fk_Illness_has_Speciality_Illness1`
```

```

FOREIGN KEY (`Illness_idIllness`)
REFERENCES `mydb`.`Illness` (`idIllness`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Illness_has_Speciality_Speciality1`
FOREIGN KEY (`Speciality_idSpeciality`)
REFERENCES `mydb`.`Speciality` (`idSpeciality`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

Создание таблицы связи таблиц «Пациент» и «Заболевание»:

```

CREATE TABLE IF NOT EXISTS `mydb`.`Patient_has_Illness` (
  `Patient_idPatient` INT NOT NULL,
  `Illness_idIllness` INT NOT NULL,
  PRIMARY KEY (`Patient_idPatient`, `Illness_idIllness`),
  INDEX `fk_Patient_has_Illness_Illness1_idx`
(`Illness_idIllness` ASC) VISIBLE,
  INDEX `fk_Patient_has_Illness_Patient1_idx`
(`Patient_idPatient` ASC) VISIBLE,
  CONSTRAINT `fk_Patient_has_Illness_Patient1`
FOREIGN KEY (`Patient_idPatient`)
REFERENCES `mydb`.`Patient` (`idPatient`)
ON DELETE NO ACTION
ON UPDATE NO ACTION,
CONSTRAINT `fk_Patient_has_Illness_Illness1`
FOREIGN KEY (`Illness_idIllness`)
REFERENCES `mydb`.`Illness` (`idIllness`)
ON DELETE NO ACTION
ON UPDATE NO ACTION)
ENGINE = InnoDB;

```

Листинг 2 запросы на получение данных

Получение списка работающих врачей:

```
SELECT * FROM Doctor;
```

Получение списка больных с некоторым заболеванием:

```
SELECT p.*, phi.name FROM Patient p, Patient_has_Illness phi
WHERE p.idPatient = phi.Patient_idPatient
AND phi.Illness_idIllness = (SELECT idIllness FROM Illness WHERE
name = ?);
```

Получение списка заболеваний диспансерных больных:

```
SELECT phi.name FROM Patient p, Patient_has_Illness phi
WHERE p.idPatient = phi.Patient_idPatient
AND phi.Illness_idIllness = (SELECT idIllness FROM Illness WHERE
name = ?);
```

Получение списка специалистов, курирующих заданное заболевание:

```
SELECT d.FIO, i.name FROM Doctor d, Illness_has_Specialty ihs,
Illness i
WHERE d.Specialty_idSpecialty = ihs.Specialty_idSpecialty
AND i.idIllness = ihs.Illness_idIllness
AND i.name = ?;
```

Получение списка больных некоторого врача:

```
SELECT d.FIO, p.FIO FROM Doctor d, Patient p WHERE d.idDoctor =
p.Doctor_idDoctor
AND d.FIO LIKE ?;
```

Получение списка больных, подлежащих осмотру в некоторый день:

```
SELECT p.*, a.date FROM Patient p, Appointment a
WHERE p.idPatient = a.Patient_idPatient
AND a.date = ?;
```