Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Южно-Уральский государственный университет»

(национальный исследовательский университет)

Институт естественных и точных наук

Факультет математики, механики и компьютерных технологий

Кафедра прикладной математики и программирования

|  |  |
| --- | --- |
|  |  |

Больница

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОЙ РАБОТЕ

по дисциплине проектирование баз данных

ЮУрГУ–010400.2023.23.ПЗ КР

|  |  |
| --- | --- |
|  | Руководитель работы,  \_\_\_\_\_\_\_\_\_\_/Т.Ю. Оленчикова  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г. |
|  | Автор работы  Студент группы ЕТ-313  \_\_\_\_\_\_\_\_\_\_\_\_\_/ К.В.Абзалимов  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г. |
|  |  |

Челябинск 2023

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования «Южно-Уральский государственный университет

(национальный исследовательский университет)»  
Институт естественных и точных наук

Факультет математики, механики и компьютерных технологий

Кафедра прикладной математики и программирования

Направление подготовки Прикладная математика и информатика

УТВЕРЖДАЮ

Заведующий кафедрой ПРИМА

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.А.Замышляева

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2022 г.

**ЗАДАНИЕ**

**на курсовую работу студента**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Абзалимов К.В.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Группа \_\_ЕТ-313\_\_\_\_

1. Дисциплина *\_\_\_\_\_Администрирование и проектирование хранилищ данных\_\_\_\_\_\_\_\_\_\_*

2. Тема работы \_\_\_\_Больница\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3. Срок сдачи студентом законченной работы *5 июня 2022 г.*

4. Перечень вопросов, подлежащих разработке:

1. разработка технического задания (цель, инфологическая модель, пользователи БД; автоматизируемые функции)
2. построение ER-модели (сущности, связи, атрибуты, функциональные зависимости);
3. построение реляционной модели БД; (переход от ER-модели к реляционной, проверка условий нормализации)
4. разработка объектов БД на стороне сервера (разграничение прав доступа, домены, таблицы, связи, триггеры, хранимые процедуры);
5. разработка интерфейса клиентского приложения для работы с БД
6. создание, отладка и тестирование программы;
7. оформление программной документации (руководство пользователя, листинг кода) и отчета по курсовой работе.

5. Календарный план

|  |  |  |
| --- | --- | --- |
| **Наименование разделов**  **курсовой работы** | **Срок выполнения**  **разделов работы** | **Отметка**  **о выполнении**  **руководителя** |
| разработка технического задания | 01.03.2022-09.03.2022 |  |
| построение ER-модели | 10.03.2022-19.03.2022 |  |
| построение реляционной модели БД | 20.03.2022-25.03.2022 |  |
| разработка объектов БД на стороне сервера | 26.03.2022-09.04.2022 |  |
| разработка интерфейса клиентского приложения для работы с БД | 10.04.2022-19.04.2022 |  |
| создание, отладка и тестирование программы | 20.04.2022-17.05.2022 |  |
| оформление программной документации и отчета по курсовой работе | 18.05.2022-05.06.2022 |  |
| защита курсовой работы | 06.06.2022-23.06.2022 |  |

Руководитель работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Т.Ю.Оленчикова

Студент \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /К.В. Абзалимов\_\_\_\_\_\_\_\_\_\_

(подпись)

ОГЛАВЛЕНИЕ

[ВВЕДЕНИЕ 4](#_gjdgxs)

[Цель работы 4](#_30j0zll)

[1 ПОСТАНОВКА ЗАДАЧИ 5](#_1fob9te)

[1.1 Инфологическая модель 5](#_3znysh7)

[1.2 Пользователи системы 5](#_2et92p0)

[1.3 Автоматизируемые функции 5](#_tyjcwt)

[2 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ 6](#_3dy6vkm)

[2.1 Построение ER-диаграммы 6](#_1t3h5sf)

[2.2 Выявление ограничений и правил поддержания целостности данных 7](#_4d34og8)

[2.2.1 Объект «Больница» 7](#_17dp8vu)

[2.2.2 Объект «Корпус» 7](#_3rdcrjn)

2.2.3 Объект «Отделение» 7

2.2.4 Объект «Палата» 7

[2.2.5 Объект «Заболевание» 7](#_35nkun2)

[2.2.6 Объект «История болезни» 8](#_1ksv4uv)

[2.2.7 Объект «Врач» 8](#_44sinio)

[2.2.8 Объект «Услуга» 8](#_2jxsxqh)

[2.2.9 Объект «Лекарство» 8](#_3j2qqm3)

[2.2.10 Объект «Рецепт» 9](#_4i7ojhp)

[2.2.11 Объект «Направление» 9](#_2xcytpi)

[2.2.12 Объект «Пациент» 9](#_3whwml4)

[2.2.13 Объект «Пребывание в палате» 9](#_3as4poj)

[2.2.13 Объект «Роль» 9](#_e656o5pdn41e)

[**2.3 Проверка модели 10**](#_1pxezwc)

[**2.4 Функциональные зависимости 10**](#_49x2ik5)

[«Больница» 10](#_2p2csry)

[«Корпус» 10](#_147n2zr)

[«Отделение» 10](#_3o7alnk)

[«Палата» 10](#_23ckvvd)

[«Заболевание» 10](#_ihv636)

[«История болезней» 10](#_32hioqz)

[«Врач» 10](#_1hmsyys)

[«Услуга» 10](#_41mghml)

[«Лекарство» 10](#_2grqrue)

[«Рецепт» 10](#_vx1227)

[«Направление» 10](#_3fwokq0)

[«Пациент» 10](#_1v1yuxt)

[«Пребывание в палате» 10](#_4f1mdlm)

[3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 11](#_2u6wntf)

[3.1 Перевод ER-модели в реляционную форму 11](#_19c6y18)

[3.1.1 Таблица hospitals («Больница») 11](#_3tbugp1)

[3.1.2 Таблица buildings («Корпус») 11](#_28h4qwu)

[3.1.3 Таблица departments («Отделения») 12](#_1mrcu09)

[3.1.4 Таблица wards («Палата») 12](#_2lwamvv)

[3.1.5 Таблица doctors («Врач») 13](#_206ipza)

[3.1.6 Таблица diseases («Заболевание») 13](#_3ygebqi)

[3.1.7 Таблица medHistories («История болезней») 14](#_sqyw64)

[3.1.8 Таблица stayinInWard («Пребывание в палате») 14](#_4bvk7pj)

[3.1.9 Таблица patients(«Пациент») 15](#_1664s55)

[3.1.10 Таблица prescriptions («Рецепт») 15](#_kgcv8k)

[3.1.11 Таблица medicines («Лекарство») 16](#_1jlao46)

[3.1.12 Таблица refferals («Направление») 16](#_2iq8gzs)

[3.1.13 Таблица services («Услуга») 16](#_3hv69ve)

[3.1.13 Таблица roles («Роль») 17](#_op8u3qwrebkc)

[3.2 Структура связей БД 17](#_4h042r0)

[3.3 Спецификация ограничений и правил поддержания целостности 18](#_2w5ecyt)

[3.4 Вывод 19](#_1baon6m)

[4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ 20](#_3vac5uf)

[4.1 Права пользователей 20](#_2afmg28)

[4.1.1 Пациент 20](#_pkwqa1)

[getWardAccount 20](#_pruh7nzhegne)

[getMedicineAccount 20](#_9cqajt81rf39)

[getServiceAccount 20](#_6az085y150ds)

[aboutPatient 20](#_c0zcf1oeqsz6)

[addDesease 20](#_tjz5tr8hhny4)

[addMedicine 20](#_h27tksjcwxj5)

[addPatient 20](#_2gx71va2qind)

[addService 20](#_deomcqpe3usn)

[addStayingInWard 20](#_wb4ung5n0g9c)

[CHECK\_PSW 20](#_8mjsvhn9vr7r)

[doctorsPatient 20](#_q4z9rq4no9ou)

[4.2.2 Врач 20](#_39kk8xu)

[getWardAccount 21](#_g2kxk9zd0y3p)

[getMedicineAccount 21](#_xn69nwthw6pv)

[getServiceAccount 21](#_ijy39p50zy)

[aboutPatient 21](#_oqgzl02u56d9)

[addDesease 21](#_md4pas8w0v1s)

[addMedicine 21](#_pbi3pg344njv)

[addPatient 21](#_sbqmbjw82wo9)

[addService 21](#_hipnwpb1598k)

[addStayingInWard 21](#_fk1rv67i14no)

[CHECK\_PSW 21](#_3qfhhrfs5094)

[doctorsPatient 21](#_vjro9nfqff3u)

[4.3.3 Руководство 21](#_1opuj5n)

[getWardAccount 21](#_p9r5nww74ejr)

[getMedicineAccount 21](#_h1th6fsm84cr)

[getServiceAccount 21](#_vm1g8cny8oi3)

[aboutPatient 21](#_mttk3c4cngw2)

[addDesease 21](#_7ppkm7tq12nd)

[addMedicine 21](#_lscokvr5shr3)

[addPatient 21](#_xjfq2ks1wz3j)

[addService 21](#_e80rpqbxc6x)

[addStayingInWard 21](#_xznfk2v1ybgf)

[CHECK\_PSW 21](#_rtjn5dw51kfb)

[doctorsPatient 21](#_uvgoo2mg4d73)

[4.2 Функции 22](#_48pi1tg)

[4.2.1 Скалярная функция getWardAccount 22](#_2nusc19)

[4.2.2 Скалярная функция getMedicineAccount 22](#_1302m92)

[4.2.3 Скалярная функция getServiceAccount 23](#_3mzq4wv)

[4.3 Триггеры 23](#_haapch)

[4.3.1 Триггеры таблицы StayingInWard 23](#_319y80a)

[4.4 Хранимые процедуры 24](#_aumnm3ahne5v)

[4.4.1 Информация о пациенте 24](#_t1zc8qsvx13x)

[4.4.2 Добавить заболевание 25](#_h91rjpbi6tox)

[4.4.3 Добавить лекарство 25](#_xgb9tqxk5cto)

[4.4.4 Добавить пациента 25](#_79tpjqizsp4j)

[4.4.5 Добавить услугу 26](#_gth2i4mkzcbz)

[4.4.6 Добавить пациента в палату 26](#_b7866esiwadf)

[4.4.7 Проверка пароля 27](#_7koyzg9dda2s)

[4.4.8 Пациента определенного доктора. 28](#_yvlv9kkmwpwq)

[**5 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ 29**](#_2dlolyb)

[5.1 Подключение к источнику данных 29](#_d8heox3vtq3p)

[5.2 Рабочая область приложения для руководства 29](#_37fu5ra3uc6k)

[5.3 Рабочая область приложения для врачей 31](#_qelkqp8wn5o0)

[**6 ОПИСАНИЕ ПРОГРАММЫ 34**](#_3r32n1rqm52i)

[6.1 Среда разработки 34](#_2r0uhxc)

[6.2 Минимальные системные требования 34](#_2j7yl94ifzha)

[6.3 Накладываемые ограничения 34](#_8u44ymtubdr5)

[**ЗАКЛЮЧЕНИЕ 35**](#_25b2l0r)

[**ЛИТЕРАТУРА 36**](#_9y7930h458w4)

# ВВЕДЕНИЕ

## Цель работы

Необходимо разработать структуру реляционной базы данных для информационной системы «Больница».

Система предназначена для отслеживания действий по лечению персонала больниц некоторого города, введение истории болезней и расчета стоимости лечения больного.

# 1 ПОСТАНОВКА ЗАДАЧИ

## 1.1 Инфологическая модель

Каждая больница города состоит из одного или нескольких корпусов. В корпусе размещается одно или несколько отделений. Отделение специализируется на лечении определенной группы заболеваний. Каждое отделение имеет некоторое количество палат на определенное число коек. У больницы имеются нормативы на длительность лечения (нахождения пациента в стационаре), а также свой набор услуг и их стоимость.

Для каждого больного ведется история болезни, которую заполняет лечащий врач. В истории болезни указываются назначенные процедуры, приобретенные услуги, предоставленные палаты и их влияние на состояние больного. Для уточнения диагноза пациент может быть направлен на консультацию в другое отделение.

Кроме того, база данных должна содержать сведения для расчета стоимости лечения: стоимость нахождения пациента в больнице, стоимость назначенных услуг и лекарств.

## 1.2 Пользователи системы

1. Врачи (В);
2. Пациенты (П);
3. Руководство (Р).

## 1.3 Автоматизируемые функции

1. Просмотр информации о лечение больных (В, Р);
2. Просмотр персональной информации о лечение больного (В, П);
3. Просмотр назначенных рецептов и направлений (П, В);
4. Просмотр информации о заполненности палат (Р);
5. Изменение данных о врачах (Р);
6. Информацию о назначенных пациентов врачу (Р).

# 2 КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ

Данный раздел посвящен проектированию и нормализации глобальной ER-модели. Строится ее графическое представление, специфицируются ограничения и правила поддержания целостности на уровне глобальной модели.

## 2.1 Построение ER-диаграммы

Выделено семь сильных сущностей: «Больница», «Пациент», «Врач», «История болезни», «Лекарство», «Услуга» и четырнадцать связей типа «один ко многим» (см. рисунок 2.1).

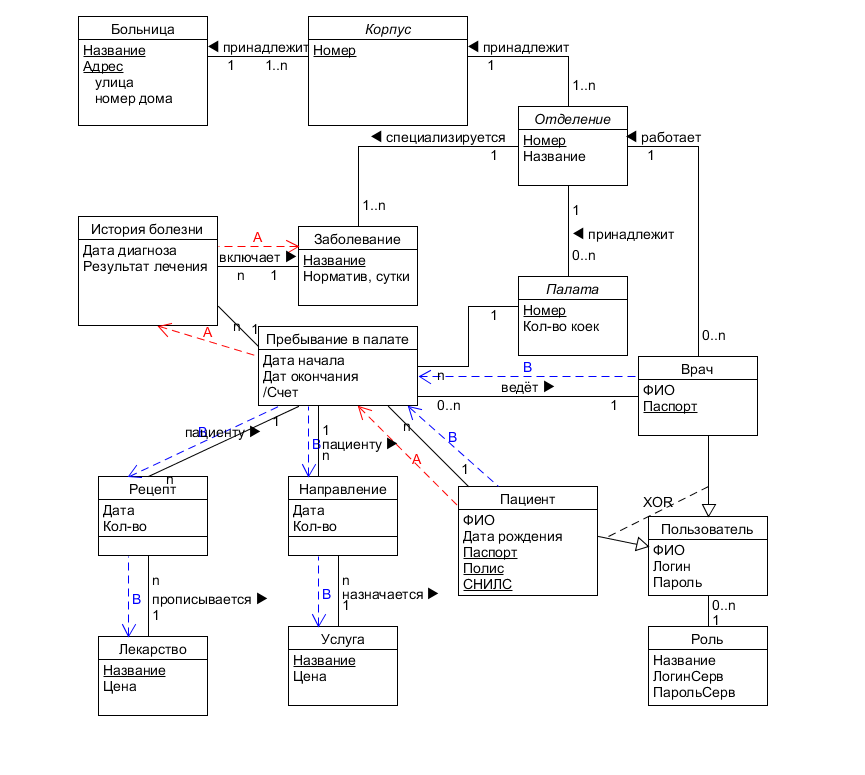


Рисунок 2.1 – ER-диаграмма

## 2.2 Выявление ограничений и правил поддержания целостности данных

Данный подраздел посвящен выявлению ограничений и правил поддержания целостности данных, которые будут размещаться в базе данных.

### 

### 2.2.1 Объект «Больница»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Название | Поле должно быть уникально. Не может быть двух больниц с одинаковым названием. |
| Адрес | Поле должно быть уникально. Не может быть двух больниц с одним адресом. Поле является составным, состоящим из атрибутов “Улица”, “Номер дома”. |

### 2.2.2 Объект «Корпус»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Номер | Поле должно быть уникально в пределах больницы. Не может быть двух корпусов с одинаковым номер в одной больнице. |

### 

### 2.2.3 Объект «Отделение»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Номер | Поле должно быть уникально в пределах больницы. Не может быть двух отделений с одинаковым номер в одной больнице. |
| Название |  |

### 2.2.4 Объект «Палата»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Номер | Поле уникальное в пределах отделения. Не может быть двух палат с одинаковым номером в одном отделении. |
| Количество коек | Количество коек >= 0 |

### 2.2.5 Объект «Заболевание»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Название | Поле уникальное. Нет заболеваний с одним и тем же названием. |
| Норматив | Количество суток на лечение. Норматив >= 0. |

### 2.2.6 Объект «История болезни»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Дата диагноза |  |
| Результат лечения |  |

### 2.2.7 Объект «Врач»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| ФИО |  |
| Паспорт | Поле уникальное. |

### 2.2.8 Объект «Услуга»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Название | Поле уникальное. |
| Цена |  |

### 

### 2.2.9 Объект «Лекарство»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Название | Поле уникальное. |
| Цена |  |

### 

### 2.2.10 Объект «Рецепт»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Дата | Дата не ранее даты начала пребывания в палате и не позднее даты окончания пребывания в палате. |
| Количество | >0 |

### 2.2.11 Объект «Направление»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Дата | Дата не ранее даты начала пребывания в палате и не позднее даты окончания пребывания в палате. |
| Количество | >0 |

### 

### 2.2.12 Объект «Пациент»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| ФИО |  |
| Дата рождения |  |
| Паспорт | Поле уникальное. |
| Полис | Поле уникальное. |
| СНИЛС | Поле уникальное. |

### 

### 

### 2.2.13 Объект «Пребывание в палате»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| Дата начала/Дата окончания | Дата начала не может быть после даты окончания. |
| Счет | Вычислимое поле. |

### 2.2.13 Объект «Роль»

|  |  |
| --- | --- |
| Имя атрибута | Ограничения |
| ЛогинСерв | Поле должно быть уникально. |
| ПарольСерв |  |
| Название Роли | Поле должно быть уникально |

# 2.3 Проверка модели

План выполнения запросов «Просмотр персональной информации о лечение больного» (A) и «Просмотр назначенных рецептов и направлений» (B) показан на рис. 2.1. Проверка этого плана показала, что ER-модель соответствует функциональным требованиям и предметной области.

# 2.4 Функциональные зависимости

Выявлены следующие функциональные зависимости в предметной области:

|  |  |
| --- | --- |
| Функциональная зависимость | Сущность |
| Название, Адрес → | «Больница» |
| Номер → | «Корпус» |
| Номер → Название | «Отделение» |
| Номер → Количество коек | «Палата» |
| Название → Норматив | «Заболевание» |
|  | «История болезней» |
| Паспорт → ФИО | «Врач» |
| Название → Цена | «Услуга» |
| Название → Цена | «Лекарство» |
|  | «Рецепт» |
|  | «Направление» |
| Паспорт → ФИО, Дата рождения, СНИЛС, Полис  СНИЛС → ФИО, Дата рождения, Паспорт, Полис  Полис → ФИО, Дата рождения, Паспорт, СНИЛС  и различные комбинации полей Паспорт, СНИЛС, Полис | «Пациент» |
|  | «Пребывание в палате» |

# 3 ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

Данный раздел посвящен проектированию реляционной SQL-модели. Здесь выполняется перевод ER-модели в реляционную форму, специфицируются ограничения и правила поддержания целостности на реляционном уровне.

## 3.1 Перевод ER-модели в реляционную форму

В ER-модели между сущностями нет связей «многие-ко-многим», есть один составной атрибут адрес, но его значения мы объединим в обычный атрибут, поэтому дополнительные таблицы не требуются. В реляционной модели каждой сущности соответствует таблица.

В разработанной реляционной форме глобальной модели используется 13 таблиц. Рассмотрим каждую из них более подробно.

### 3.1.1 Таблица hospitals («Больница»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| hospital\_ID | int | identity | + |  | + | + | - |
| name | nvarchar(100) | - | + |  | + | - | - |
| address | nvarchar(100) | - | + |  | + | - | - |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity. Кандидаты на первичный ключ, уникальные поля «Название» и «Адрес», не используются из-за их символьного формата.

### 3.1.2 Таблица buildings («Корпус»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| building\_ID | int | identity | + |  | + | + | - |
| buildNumber | tinyint | - | + | индекс | + | - | - |
| hospital\_ID | int | - | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

Поле buildNumber должно быть уникально в пределах больницы, поэтому поля (buildNumber, hospital\_ID) уникальны, также создаем индекс по этим же полям.

### 3.1.3 Таблица departments («Отделения»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| department\_ID | int | identity | + |  | + | + | - |
| depName | nvarchar(50) | - | - |  | + | - | - |
| depNumber | tinyint | - | + | >0  индекс | + | - | - |
| bulding\_ID | int | - | - | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

Поле depNumber должно быть уникально в пределах корпуса, поэтому поля (number, building\_ID) уникальны, также создаем индекс по этим же полям.

### 3.1.4 Таблица wards («Палата»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| ward\_ID | int | identity | + |  | + | + | - |
| bedsCount | smallint | - | - | >0 | + | - | - |
| wardNumber | smallint | - | + | >0  индекс | + | - | - |
| department\_ID | int | - | - | - | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

ПолеwardNumber должно быть уникально в пределах отделения, поэтому поля (wardNumber, department\_ID) уникальны, также создаем индекс по этим же полям.

### 3.1.5 Таблица doctors («Врач»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| doctor\_ID | int | identity | + |  | + | + | - |
| pass | nchar(11) | - | + | цифры | + | - | - |
| FIO | nvarchar(50) | - | - | буквы | + | - | - |
| department\_ID | int | - | - |  | - | - | + |
| doctorLogin | nvarchar(60) | - | + | - | - | - | - |
| doctorPass | nchar(256) | - | - | - | - | - | - |
| role\_ID | int | - | - | - | + | - | + |

### 

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity. Кандидат на первичный ключ, уникальное поле «Паспорт», не используются из-за их символьного формата.

### 

### 3.1.6 Таблица diseases («Заболевание»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| desease\_ID | int | identity | + |  | + | + | - |
| name | nvarchar(50) | - | + |  | + | - | - |
| standartCountDay | tinyint | - | - | >0 | + | - | - |
| department\_ID | int | - | - |  | - | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity. Кандидат на первичный ключ, уникальное поле «Название», не используются из-за их символьного формата.

### 3.1.7 Таблица medHistories («История болезней»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| medHistory\_ID | int | identity | + |  | + | + | - |
| date | date | getdate() | - |  | + | - | - |
| result | text | - | - |  | + | - | - |
| desease\_ID | int | - | - |  | + | - | + |

## 

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.8 Таблица stayinInWard («Пребывание в палате»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| stayingInWard\_ID | int | identity | + |  | + | + | - |
| dateStart | date | - | + | dateStart<  dateEnd | + | - | - |
| dateEnd | date | - | - | + | - | - |
| account | money | - | - | вычислимое | - | - | - |
| ward\_ID | int | - | - |  | + | - | + |
| doctor\_ID | int | - | - |  | - | - | + |
| patient\_ID | int | - | - |  | + | - | + |
| medHistory\_ID | int | - | - |  | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

Поле account является вычислимым, для его вычисления будет использоваться скалярная функция calcAccount.

Для сохранения ограничения по атрибутам dateStart и dateEnd будет использоваться триггер startEnd.

### 3.1.9 Таблица patients(«Пациент»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| patient\_ID | int | identity | + |  | + | + | - |
| pass | nchar(10) | - | + | цифры | + | - | - |
| polis | nchar(16) | - | + | цифры | + | - | - |
| snils | nchar(11) | - | + | цифры | + | - | - |
| dateOfBirth | date | - | - | <getdate() | + | - | - |
| FIO | nvarchar(50) | - | - | буквы | + | - | - |
| address | nvarchar(100) | - | - |  | + | - | - |
| patientLogin | nvarchar(60) | - | + | - | - | - | - |
| patientPassword | nchar(256) | - | - | - | - | - | - |
| role\_ID | int | - | - | - | + | - | + |

## 

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.10 Таблица prescriptions («Рецепт»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| prescription\_ID | int | identity | + |  | + | + | - |
| date | date | - | + |  | + | - | - |
| count | tinyint | - | + | >0 | + | - | - |
| stayInWard\_ID | int | - | - |  | + | - | + |
| medicine\_ID | int | - | - |  | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.11 Таблица medicines («Лекарство»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| medicine\_ID | int | identity | + |  | + | + | - |
| name | nvarchar(60) | - | + |  | + | - | - |
| cost | money | - | - | >0 | + | - | - |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.12 Таблица refferals («Направление»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| refferals\_ID | int | identity | + |  | + | + | - |
| date | date | - | - |  | + | - | - |
| count | int | - | - |  | + | - | - |
| stayInWard\_ID | date | - | - |  | + | - | + |
| service\_ID | int | - | - |  | + | - | + |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.13 Таблица services («Услуга»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| service\_ID | int | identity | + |  | + | + | - |
| name | nvarchar(50) | - | + |  | + | - | - |
| cost | money | - | - | >0 | + | - | - |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

### 3.1.13 Таблица roles («Роль»)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Название | Тип данных | Значение по умолчанию | Уникальность | Ограничения | Обязательность | ПК | ВК |
| role\_ID | uniqueidentifier | identity | + |  | + | + | - |
| roleName | nvarchar(100) | - | + |  | + | - | - |
| servLogin | nvarchar(60) | - | + |  | + | - | - |
| servPass | nchar(256) | - | - |  | + | - | - |

Для идентификации записей введен суррогатный первичный ключ со значением по умолчанию identity.

## 3.2 Структура связей БД

В этом подразделе будет описана структура связей между таблицами построенной базы данных. Для большей наглядности эти связи будут представлены с помощью диаграммы (см. рисунок 3.1).

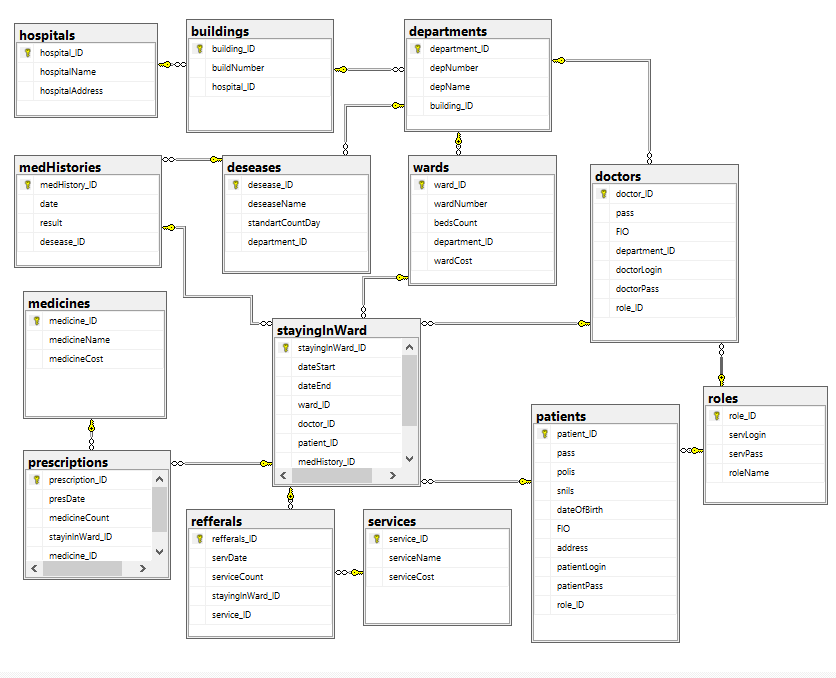


Рисунок 3.1 – Диаграмма «Hospitals»

## 3.3 Спецификация ограничений и правил поддержания целостности

В данном подразделе учитываются ограничения и правила поддержания целостности, выявленные на предыдущих этапах. Новые ограничения и правила обнаружены не были.

3.4 Проверка условий нормализации.

Проверка проводится отдельно для каждой таблицы на соответствие нормальным формам.

Все таблицы БД находятся в НФБК так как:

* 1НФ – значения всех атрибутов атомарны и не повторяются;
* 2НФ – отсутствуют функциональные зависимости от части первичного ключа;
* 3НФ – отсутствуют транзитивные зависимости от первичного ключа
* НФБК – отсутствуют функциональные зависимости части первичного ключа от неключевых атрибутов.

Дальнейшая нормализация БД не требуется.

## 3.4 Вывод

В данном разделе была спроектирована реляционная модель БД, для чего был сделан перевод глобальной ER-модели в реляционную форму.

# 4 ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

В этом разделе будут подробно описаны используемые хранимые процедуры, табличные и скалярные функции, триггеры, а также права, предоставленные различным группам пользователей.

## 4.1 Права пользователей

Защита базы данных от несанкционированного доступа осуществляется с помощью создания ролей Пациента, Врача и Руководства, которые имеют определенные права доступа к таблицам, отдельным столбцам таблиц базы, скалярным и табличным функциям и хранимым процедурам.

Рассмотрим каждую роль и предоставленные ей права подробнее.

### 4.1.1 Пациент

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Права пользователя | | | |
| Вставка | Выборка | Обновление | Удаление |
| hospitals | Запрещено | Запрещено | Запрещено | Запрещено |
| buildigs | Запрещено | Запрещено | Запрещено | Запрещено |
| departments | Запрещено | Запрещено | Запрещено | Запрещено |
| wards | Запрещено | Запрещено | Запрещено | Запрещено |
| deseases | Запрещено | Запрещено | Запрещено | Запрещено |
| medHistories | Запрещено | Разрешено | Запрещено | Запрещено |
| doctors | Запрещено | Запрещено | Запрещено | Запрещено |
| patients | Запрещено | Разрешено | Запрещено | Запрещено |
| stayingInWard | Запрещено | Разрешено | Запрещено | Запрещено |
| prescriptions | Запрещено | Разрешено | Запрещено | Запрещено |
| medicines | Запрещено | Разрешено | Запрещено | Запрещено |
| refferals | Запрещено | Разрешено | Запрещено | Запрещено |
| services | Запрещено | Разрешено | Запрещено | Запрещено |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Права | Хранимые процедуры, функции | | | | | | | | | | |
| getWardAccount | getMedicineAccount | getServiceAccount | aboutPatient | addDesease | addMedicine | addPatient | addService | addStayingInWard | CHECK\_PSW | doctorsPatient |
| Выполнение | + | + | + | + | - | + | - | + | + | + | - |

### 4.2.2 Врач

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Права пользователя | | | |
| Вставка | Выборка | Обновление | Удаление |
| hospitals | Запрещено | Запрещено | Запрещено | Запрещено |
| buildigs | Запрещено | Запрещено | Запрещено | Запрещено |
| departments | Запрещено | Запрещено | Запрещено | Запрещено |
| wards | Запрещено | Запрещено | Запрещено | Запрещено |
| deseases | Запрещено | Запрещено | Запрещено | Запрещено |
| medHistories | Разрешено | Разрешено | Разрешено | Запрещено |
| doctors | Запрещено | Запрещено | Запрещено | Запрещено |
| patients | Разрешено | Разрешено | Разрешено | Запрещено |
| stayingInWard | Разрешено | Разрешено | Разрешено | Запрещено |
| prescriptions | Разрешено | Разрешено | Запрещено | Запрещено |
| medicines | Запрещено | Разрешено | Запрещено | Запрещено |
| refferals | Разрешено | Разрешено | Запрещено | Запрещено |
| services | Запрещено | Разрешено | Запрещено | Запрещено |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Права | Хранимые процедуры, функции | | | | | | | | | | |
| getWardAccount | getMedicineAccount | getServiceAccount | aboutPatient | addDesease | addMedicine | addPatient | addService | addStayingInWard | CHECK\_PSW | doctorsPatient |
| Выполнение | + | + | + | + | - | - | + | - | + | + | + |

### 4.3.3 Руководство

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Таблица | Права пользователя | | | |
| Вставка | Выборка | Обновление | Удаление |
| hospitals | Разрешено | Разрешено | Разрешено | Разрешено |
| buildigs | Разрешено | Разрешено | Разрешено | Разрешено |
| departments | Разрешено | Разрешено | Разрешено | Разрешено |
| wards | Разрешено | Разрешено | Разрешено | Разрешено |
| deseases | Разрешено | Разрешено | Разрешено | Разрешено |
| medHistories | Запрещено | Разрешено | Запрещено | Запрещено |
| doctors | Разрешено | Разрешено | Разрешено | Разрешено |
| patients | Запрещено | Разрешено | Запрещено | Запрещено |
| stayingInWard | Запрещено | Разрешено | Запрещено | Запрещено |
| prescriptions | Запрещено | Разрешено | Запрещено | Запрещено |
| medicines | Разрешено | Разрешено | Разрешено | Разрешено |
| refferals | Запрещено | Разрешено | Запрещено | Запрещено |
| services | Разрешено | Разрешено | Разрешено | Разрешено |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Права | Хранимые процедуры, функции | | | | | | | | | | |
| getWardAccount | getMedicineAccount | getServiceAccount | aboutPatient | addDesease | addMedicine | addPatient | addService | addStayingInWard | CHECK\_PSW | doctorsPatient |
| Выполнение | + | + | + | - | + | + | - | + | - | + | - |

## 4.2 Функции

Реализовано 3 скалярных функций, основная задача которых – облегчить работу с запросами к БД.

### 4.2.1 Скалярная функция getWardAccount

Входные параметры функции – stayingInWard\_ID из таблицы stayingInWard. Возвращает счет за пребывание в палате(см. листинг 4.5)

-- Листинг 4.5 скалярная функция getWardAccount

CREATE FUNCTION dbo.getWardAccount(@stayingInWard\_ID int)

RETURNS money AS

BEGIN

DECLARE @res money

SET @res = (SELECT (DATEDIFF(DAY, sw.dateStart, ISNULL(sw.dateEnd, GETDATE())))\*w.wardCost

FROM wards w, stayingInWard sw

WHERE sw.stayingInWard\_ID = @stayingInWard\_ID

AND sw.ward\_ID = w.ward\_ID)

RETURN @res

END

### 4.2.2 Скалярная функция getMedicineAccount

Входные параметры функции – stayingInWard\_ID из таблицы stayingInWard. Возвращает счет за лекарства (см. листинг 4.5)

-- Листинг 4.5 скалярная функция getMedicineAccount

CREATE FUNCTION dbo.getMedicineAccount(@stayingInWard\_ID int)

RETURNS money AS

BEGIN

DECLARE @res money

SET @res = (SELECT SUM(m.medicineCost\*p.medicineCount)

FROM prescriptions p, medicines m

WHERE p.stayinInWard\_ID = @stayingInWard\_ID

AND p.medicine\_ID = m.medicine\_ID)

RETURN @res

END

### 4.2.3 Скалярная функция getServiceAccount

Входные параметры функции – stayingInWard\_ID из таблицы stayingInWard. Возвращает счет за процедуры (см. листинг 4.5)

-- Листинг 4.5 скалярная функция getServiceAccount

CREATE FUNCTION dbo.getMedicineAccount(@stayingInWard\_ID int)

RETURNS money AS

BEGIN

DECLARE @res money

SET @res = (SELECT SUM(m.medicineCost\*p.medicineCount)

FROM prescriptions p, medicines m

WHERE p.stayinInWard\_ID = @stayingInWard\_ID

AND p.medicine\_ID = m.medicine\_ID)

### 

## 4.3 Триггеры

Всего реализовано 2 триггера: 1 триггер для таблицы «StayingInWard».

### 4.3.1 Триггеры таблицы StayingInWard

Триггер не допускает, чтобы пациента с болезнью, принадлежащей одному отделению, не был в палате, принадлежащей другому отделению.

-- Листинг 4.5 триггер stInWardMatchingWardAndDesease

CREATE TRIGGER stInWardMatchingWardAndDesease

ON stayingInWard AFTER INSERT, UPDATE

AS

if @@ROWCOUNT = 0

RETURN

IF EXISTS (

SELECT dpr.department\_ID

FROM inserted i, wards w, medHistories md, deseases d, departments dpr

WHERE i.medHistory\_ID = md.medHistory\_ID

AND md.desease\_ID = d.desease\_ID

AND i.ward\_ID = w.ward\_ID

AND w.department\_ID != d.department\_ID

)

BEGIN

ROLLBACK TRANSACTION

raiserror('отделение палаты и отделение заболевания не совпадают', 16, 1)

END

Триггер не допускает, чтобы пациента лечил врач, принадлежащий одному отделению, не был в палате, принадлежащей другому отделению.

-- Листинг 4.5 триггер stInWardMatchingWardAndDoctor

CREATE TRIGGER stInWardMatchingWardAndDoctor

ON stayingInWard AFTER INSERT, UPDATE

AS

if @@ROWCOUNT = 0

RETURN

IF EXISTS (

SELECT dpr.department\_ID

FROM inserted i, wards w, doctors d, departments dpr

WHERE i.doctor\_ID = d.doctor\_ID

AND i.ward\_ID = w.ward\_ID

AND w.department\_ID != d.department\_ID

)

BEGIN

ROLLBACK TRANSACTION

raiserror('отделение палаты и отделение врача не совпадают', 16,1)

END

## 4.4 Хранимые процедуры

### 4.4.1 Информация о пациенте

CREATE PROCEDURE [dbo].[aboutPatient]

@patient\_ID int

AS

BEGIN

SELECT sw.stayingInWard\_ID as 'Номер', p.FIO as 'ФИО Пациента', p.pass as 'Паспорт', d.FIO as 'ФИО Доктора', w.wardNumber as 'Номер палаты', sw.dateStart as 'Дата поступления', sw.dateEnd as 'Дата выписики', md.result as 'История болезни'

FROM stayingInWard sw, wards w, doctors d, medHistories md, patients p

WHERE sw.patient\_ID = p.patient\_ID

AND w.ward\_ID = sw.ward\_ID

AND sw.doctor\_ID = d.doctor\_ID

AND sw.patient\_ID = @patient\_ID

AND w.ward\_ID = sw.ward\_ID

AND sw.medHistory\_ID = md.medHistory\_ID

END

### 4.4.2 Добавить заболевание

CREATE PROCEDURE dbo.addDesease

@deseaseName nvarchar(50),

@standartCountDay int,

@department\_ID int,

@desease\_ID int out

AS

BEGIN

INSERT INTO deseases(deseaseName, standartCountDay, department\_ID)

VALUES (@deseaseName, @standartCountDay, @department\_ID)

SET @desease\_ID = SCOPE\_IDENTITY()

END

### 4.4.3 Добавить лекарство

CREATE PROCEDURE dbo.addMedicine

@medicineName nvarchar(50),

@medicineCost money,

@medicine\_ID int out

AS

BEGIN

INSERT INTO medicines(medicineName, medicineCost)

VALUES (@medicineName, @medicineCost)

SET @medicine\_ID = SCOPE\_IDENTITY()

END

### 4.4.4 Добавить пациента

CREATE PROCEDURE dbo.addPatient

@pass nchar(10),

@polis nchar(16),

@snils nvarchar(11),

@dateOfBirth date,

@FIO nvarchar(50),

@address nvarchar(100),

@patientLogin nvarchar(60),

@patientPassword nvarchar(60)

AS

BEGIN

INSERT INTO patients(pass, polis, snils, dateOfBirth, FIO, address, patientLogin, patientPass)

VALUES (@pass, @polis, @snils, @dateOfBirth, @FIO, @address, @patientLogin, @patientPassword)

END

### 4.4.5 Добавить услугу

CREATE PROCEDURE dbo.addService

@serviceName nvarchar(50),

@serviceCost money,

@service\_ID int out

AS

BEGIN

INSERT INTO services(serviceName, serviceCost)

VALUES (@serviceName, @serviceCost)

SET @service\_ID = SCOPE\_IDENTITY()

END

### 4.4.6 Добавить пациента в палату

CREATE PROCEDURE dbo.addStayingInWard

@dateStart date,

@wardNumber int,

@doctor\_ID int,

@patient\_ID int,

@result text,

@desease nvarchar(50)

AS

BEGIN

DECLARE @desease\_ID int, @medHistories\_ID int, @ward\_ID int

SELECT @desease\_ID = d.desease\_ID

FROM deseases d

WHERE d.deseaseName = @desease

INSERT INTO medHistories(date, result, desease\_ID)

VALUES (GETDATE(), @result, @desease\_ID)

SET @medHistories\_ID = SCOPE\_IDENTITY()

SELECT @ward\_ID = w.ward\_ID

FROM wards w, doctors dc

WHERE w.wardNumber = @wardNumber

AND dc.doctor\_ID = @doctor\_ID

AND dc.department\_ID = w.department\_ID

INSERT INTO stayingInWard(dateStart, ward\_ID, doctor\_ID, patient\_ID, medHistory\_ID)

VALUES (@dateStart, @ward\_ID, @doctor\_ID, @patient\_ID, @medHistories\_ID)

END

### 4.4.7 Проверка пароля

CREATE PROCEDURE [dbo].[CHECK\_PSW]

@login nvarchar(60),

@psw nvarchar(60),

@serverLog nvarchar(60) out,

@serverPsw nvarchar(60) out,

@code int out,

@FIO nvarchar(50) out

AS

BEGIN

DECLARE @rowCount\_N int;

SELECT @code = 0, @serverLog = r.servLogin, @serverPsw = r.servPass, @FIO = 'Руководитель'

FROM roles r

WHERE r.servLogin = @login and r.servPass = @psw

SET @rowCount\_N = @@ROWCOUNT;

IF @rowCount\_N = 0

BEGIN

SELECT @code = d.doctor\_ID, @serverLog = r.servLogin, @serverPsw = r.servPass, @FIO = d.FIO

FROM roles r INNER JOIN doctors d ON r.role\_ID=d.role\_ID

WHERE d.doctorLogin = @login and d.doctorPass= @psw

SET @rowCount\_N = @@ROWCOUNT;

END

IF @rowCount\_N = 0

BEGIN

SELECT @code = p.patient\_ID, @serverLog = r.servLogin, @serverPsw = r.servPass, @FIO = p.FIO

FROM roles r INNER JOIN patients p ON r.role\_ID=p.role\_ID

WHERE p.patientLogin = @login and p.patientPass = @psw;

SET @rowCount\_N = @@ROWCOUNT;

END

IF @rowCount\_N = 0

BEGIN

RAISERROR ('Неверен логин или пароль!!!', 16, 1);

END

END

### 4.4.8 Пациента определенного доктора.

CREATE PROCEDURE dbo.doctorsPatient

@doctor\_ID int

AS

BEGIN

SELECT p.patient\_ID as 'ID', p.FIO as 'ФИО', p.pass as 'Паспорт',

p.snils as 'СНИЛС', p.polis as 'ПОЛИС', p.dateOfBirth as 'Дата рождения',

p.address as 'Адрес'

FROM patients p, stayingInWard sw, doctors d

WHERE p.patient\_ID = sw.patient\_ID

AND d.doctor\_ID = sw.doctor\_ID

AND d.doctor\_ID = @doctor\_ID

END

# 5 ПРОЕКТИРОВАНИЕ ПРИЛОЖЕНИЯ

Данный раздел посвящен описанию приложения, реализующего описанные ранее автоматизированные функции.

## 5.1 Подключение к источнику данных

Работа с программой начинается с подключения к SQL серверу, хранящему спроектированную ранее базу данных. Для подключения к серверу используется форма «Авторизация» (см. рисунок 5.1).

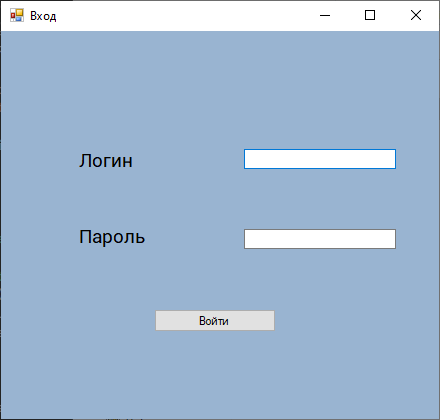


Рисунок 5.1 – Авторизация пользователя

## 5.2 Рабочая область приложения для руководства

На рисунках 5.2 и 5.3 изображены вкладки на которых руководитель может добавлять, удалять и обновлять данные, названия, которых указаны в меню.

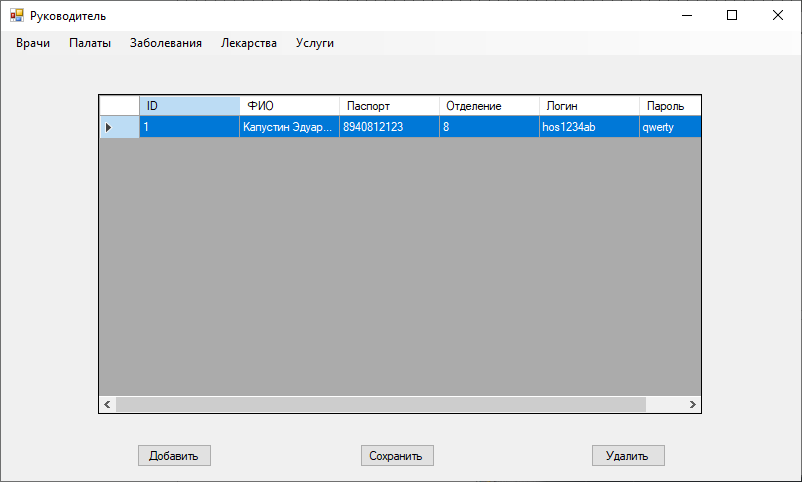


Рисунок 5.2 – Форма для работы руководителя 1

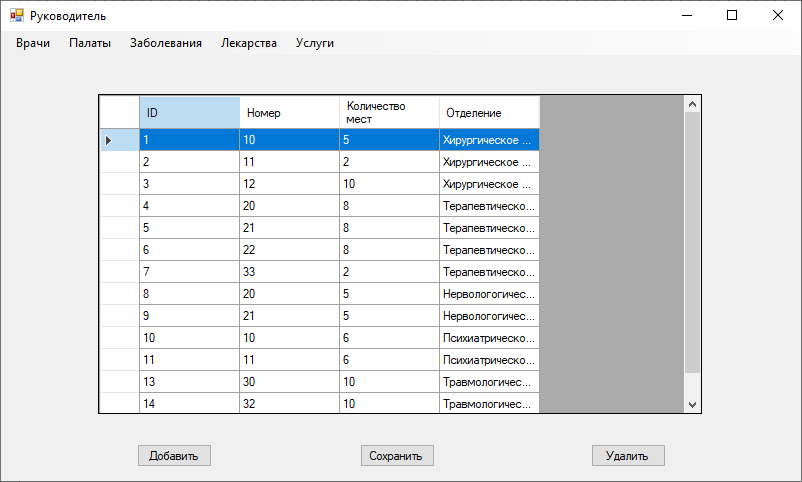


Рисунок 5.3 – Форма для работы руководителя 2

## 5.3 Рабочая область приложения для врачей

На рисунке 5.3 расположена таблица с основными данными о пациентах текущего доктора. С помощью кнопки “Новый пациент” можно добавить пациента при этом открывается форма с рисунка 5.4. Кнопка “Назначить пациента” позволяет назначить пациента выделенной строки, переходя на форму с рисунка 5.5. И кнопка “Информация о пациенте” позволяет просмотреть более подробную информацию о пациенте выделенной строки, после нажатия открывается форма рисунка 5.6.

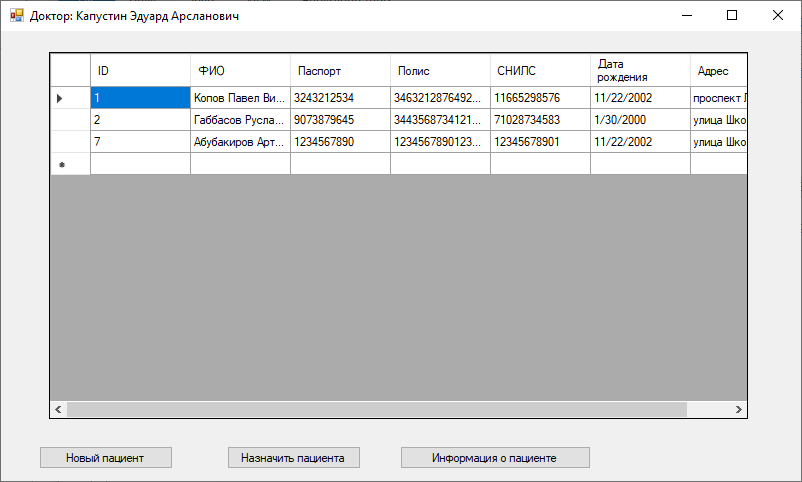


Рисунок 5.3 – Форма для работы доктора

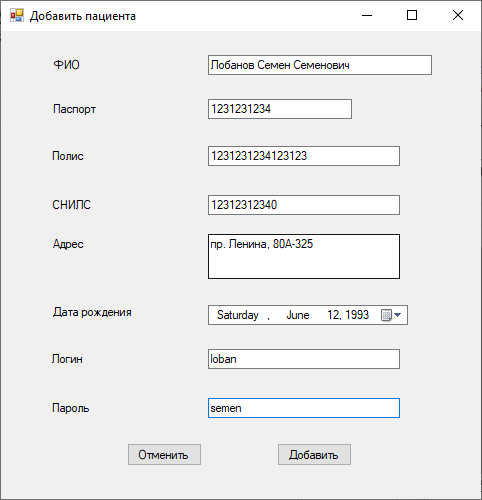


Рисунок 5.3 – Форма для добавления пациента

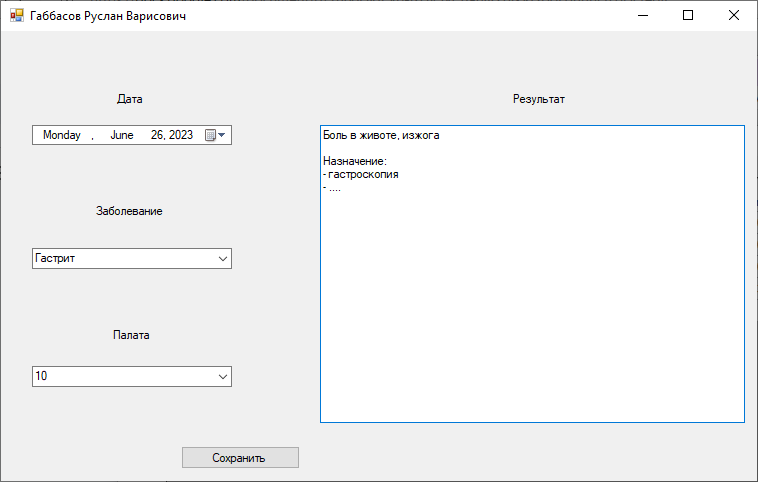


Рисунок 5.3 – Форма для добавления пациента в палату и заполнения истории

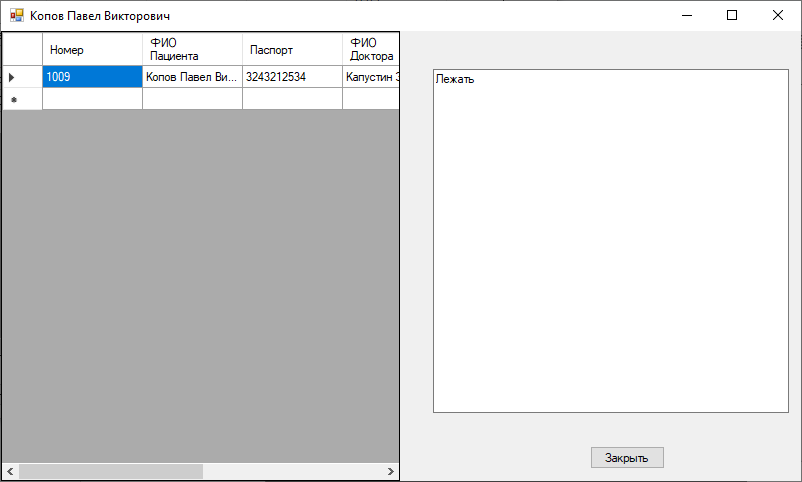


Рисунок 5.3 – Форма для обзора текущих пациентов у определенного доктора

# 6 ОПИСАНИЕ ПРОГРАММЫ

## 6.1 Среда разработки

Приложение написано в Visual Studio 2008 на C#. Для создания базы данных использован Microsoft SQL Server MS 2008.

## 6.2 Минимальные системные требования

* Наличие не менее 256 мегабайт оперативной памяти;
* Наличие не менее 50 мегабайт свободного места на жестком диске;
* Наличие операционной системы не ниже Windows7;
* Наличие установленного .NET Framework не ниже версии 3.5;
* Наличие подключения к SQL серверу.

## 6.3 Накладываемые ограничения

За исключением системных требований никаких других ограничений на программу не накладываются.

# ЗАКЛЮЧЕНИЕ

В результате проделанной работы выполнены поставленные задачи, а именно:

разработана структура реляционной базы данных для информационной системы «Больница», а также спроектировано приложение для работы с этой системой.

# ЛИТЕРАТУРА

1. Нейгел, К. C# 2008 и платформа .NET 3.5 для профессионалов / К. Нейгел [и др.] – М.: ООО “И.Д. Вильямс”, 2009. – 1392 с.
2. Евсеева, О.Н. Работа с базами данных на языке C#: учебное пособие / О.Н. Евсеева, А.Б. Шамшев. – Ульяновск, 2009. – 171 с.