**Государственное бюджетное профессиональное**

**образовательное учреждение Астраханской области**

**«Астраханский колледж вычислительной техники»**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **К защите допустить:** | | | | | | | |
| **Старший методист УМС** | | | | | | | |
|  |  | | | **Л.О.Земцова** | | | |
| **«** |  | **»** |  | | **20** |  | **г.** |

**курсовая работа**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| по МДК | 04.01 | Технология разработки и защиты баз данных | | | | | | |
|  | | | | | | | | |
|  | | | 6 семестр | | | | | |
| Тема курсовой работы | | | Разработка БД «Библиотека» | | | | | |
|  | | |  | | | | | |
|  | | |  | | | | | |
| Аквт.09.02.07.Кр00.04ПЗ | | | | | | | | |
| Листов: | | | | 33 |  | | | |
|  | |  | | | | | | |
| Разработчик | |  | | | | Студент гр. | | ПБ-33 |
|  | |  | | | | Гурьянов Д.В. | | |
|  | |  | | | |  |  | |
| Руководитель | |  | | | | Преподаватель | | |
|  | |  | | | | Мулеев Р.Р. | | |

**2022**

**Государственное бюджетное профессиональное образовательное учреждение**

**Астраханской области «Астраханский колледж вычислительной техники»**

|  |
| --- |
| УТВЕРЖДЕНО |
| Цикловой комиссией |
| специальности 09.02.07 |
| ПЦК \_\_\_\_\_\_\_\_\_ Ю.С. Андрианова |
| «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2022 г. |
| Специальность | *09.02.07 Информационные системы и программирование* | | |
| МДК 04.01 | *Технология разработки и защиты баз данных* | | |
| Группа | *ПБ– 33* | | |
| **ЗАДАНИЕ** | | | |
| **на курсовое проектирование** | | | |
| ***Гурьянову Денису Валерьевичу*** | | | |
| (фамилия, имя, отчество студента) | | | |
| Тема курсового проекта | | ***Разработка БД*** | |
| ***«Больница»*** | | | |

Содержание пояснительной записки курсового проекта

Содержание

Введение  
1 Проектирование базы данных

1.1 Анализ и описание предметной области информационной системы

1.2 Концептуальная модель

1.3 Инфологическое моделирование

1.4 Логическая модель

1.5 Выбор СУБД

2 Реализация базы данных

2.1 Физическое проектирован

2.2 Написание исходного кода БД

2.3 Написание запросов, функций и хранимых процедур

Заключение

Приложения А

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Дата выдачи задания | « |  | » |  | |  | 20 |  | г. |
| Руководитель |  | | | | Р.Р. Мулеев | | | | |
| Студент |  | | | | Д.Д. Кондрашов | | | | |

**Аннотация**

Тема курсового проекта: Разработка БД «Больница».

Представленная курсовая работа состоит из введения, двух глав, заключения.

В первой части работы рассматривается подробное описание предметной области «Больница». Дано описание проектирования инфологической и логической модели базы данных. В следующей главе описывается создание базы данных по описанным моделям с приведением кода.

Ключевые слова: база данных, больница.

**Содержание**

[Введение 5](#_Toc105285245)

[1 Проектирование базы данных 6](#_Toc105285246)

[1.1 Анализ и описание предметной области информационной системы 6](#_Toc105285247)

[1.1.1 Бизнес модель предприятия 6](#_Toc105285248)

[1.1.2 Пользователи системы 6](#_Toc105285249)

[1.1.3 Бизнес-процесс 7](#_Toc105285250)

[1.2 Концептуальная модель 7](#_Toc105285251)

[1.3 Инфологическое моделирование 9](#_Toc105285252)

[1.4 Построение логической модели 10](#_Toc105285253)

[1.5 Выбор СУБД 13](#_Toc105285254)

[2 Реализация базы данных 14](#_Toc105285255)

[2.1 Физическое проектирование 14](#_Toc105285256)

[2.2 Написание исходного кода БД 18](#_Toc105285257)

[2.3 Написание запросов, функций и хранимых процедур 21](#_Toc105285258)

[2.3.1 Общие сведения 21](#_Toc105285259)

[2.3.2 Хранимые процедуры 21](#_Toc105285260)

[2.3.3 Запросы на заполнение таблиц 25](#_Toc105285261)

[2.3.4 Триггеры 28](#_Toc105285262)

[2.3.5 Хранимые процедуры с запросами 29](#_Toc105285263)

[Заключение 32](#_Toc105285264)

**Введение**

База данных - это организованная структура, предназначенная для хранения информации. С понятием базы данных тесно связано понятие системы управления базой данных. Такая система должна:

* обеспечивать получение общих и/или детализированных отчетов по итогам работы;
* позволять легко определять тенденции изменения важнейших показателей;
* обеспечивать получение информации, критической по времени, без существенных задержек;
* выполнять точный и полный анализ данных.

Целью данной курсовой работы является анализ предметной области - «Больница», проектирование и реализация учебной базы данных в указанной предметной области. В качестве СУБД для реализации базы данных была использована СУБД реляционного типа - MS SQL Server. В рамках курсовой работы были поставлены следующие задачи:

* анализ предметной области магазин «Больница»;
* проектирование БД в исследуемой предметной области (включающее проектирование структуры таблиц, ключевых полей и схемы данных);
* приобретение навыков по созданию запросов различных типов.

1. **Проектирование базы данных**
   1. **Анализ и описание предметной области информационной системы**
      1. **Бизнес модель предприятия**

Бизнес-модель больницы представляет собой предприятие/здание(заведение) которое предоставляет медицинскую помощь своим клиентам. Оказание помощи может происходить по средству регистрации клиента на регистрационной и записи к определенному врачу. После записи клиента на услугу определенного врача(врачей), клиент получает информацию о месте проведения – в каком отделе и в каком кабинете, и времени. Также после посещения клиентом платных услуг, клиенту выдается квитанция и генерирует отчет по проведению услуги. Во время проведения услуги врач вносит(обновляет) записи о клиенте, в его медицинскую карточку. В карточке отображаются все болезни какие были у клиента во время посещений поликлиники. Больница имеет множество отделов(отделений), и каждый клиент имеет возможность попасть в любой(е). Каждый отдел(отделение) представляет собой множество услуг, которые выполняются именно в этом отделе(отделение). После того как клиент выписывается из больницы, его заносят в архив, в котором отображается какой клиент выписался и когда. Больница имеет возможность положить клиента в стационар либо же провести операцию. После каждой операции, клиенту присваиваться статус в каком состоянии от находиться. Если у клиента курс лечения будет длительным, то ему могут предложить лечь в палату. Палаты предоставляют из себя комнаты, в которых могут находиться люди только одного пола(гендера).

* + 1. **Пользователи системы**

Анализ предметной области привел к тому, что будут следующие пользователи:

* администратор – в случае необходимости, когда требуется вмешательство со стороны (уладить какие-то конфликты). Администратору предоставлен доступ к просмотру, а также и редактированию личной информации клиента;
* директор – не учувствует в больнице как какая-то рабочая сила, а является руководителем компании. В БД может просматривать отчеты, генерировать новые услуги, добавлять новых сотрудников и т.д;
* врач – может просматривать личную информацию о клиенте и историю его болезни. Также может вносить данные в некоторые таблицы, которые относятся к клиенту, такие как: статус; отчет; медицинская карта;
* приемная – имеет доступ к личной информации клиента, может записывать клиента на прием к врачу.
  + 1. **Бизнес-процесс**

В рамках данной работы был выделен основной бизнес-процесс, деятельность регистратуры.

Этот процесс включает в себя:

учет поступления пациентов. Учет включает в себя создание медицинской карты если таковой не имеется, куда вносятся данные о клиенте, запись на прием к врачу и консультацию в общем;

учет проведенного лечения. После того как клиент выписывается, ему могут выдать квитанцию либо же больничный. В данном документе предоставлены все услуги, на которые ходил клиент;

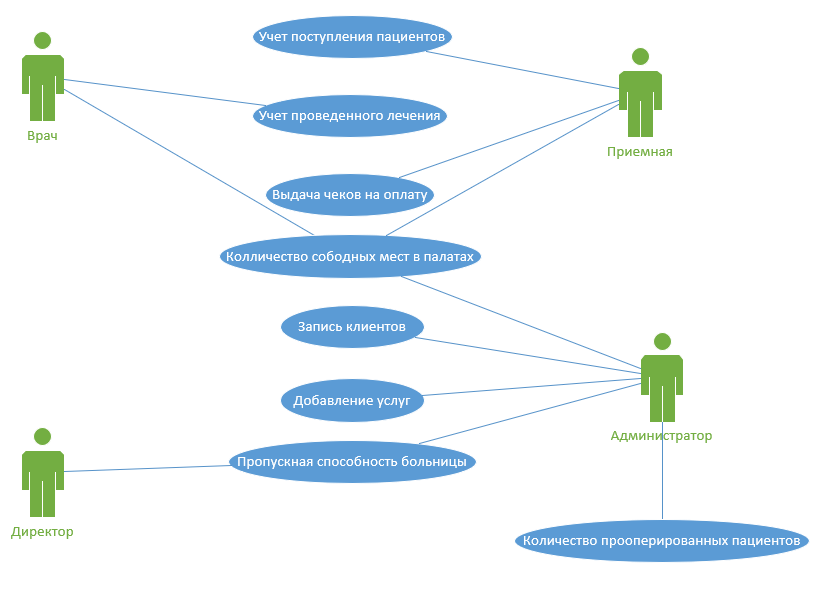
учет платных услуг с выдачей счетов на оплату. В больнице помимо бесплатных услуг есть платные. При посещении платной услуги, клиент обязан заплатить за ее выполнение, до посещения или после. После проведенного лечения по платным услугам в больнице предусмотрена выдача чеков, в которых указываются какие услуги были пройдены и в каких количествах;

ведение архива выписанных пациентов. В медицинской карте клиента, есть отдельные записи о том, когда клиент пришел на больничный, и когда он выписался. Так как такового архива в больнице нет, из этих второстепенных данных складывается архив.

* 1. **Концептуальная модель**

Диаграмма вариантов использования - это исходное концептуальное представление или концептуальная модель системы в процессе ее проектирования и разработки. Создание диаграммы вариантов использования имеет следующие цели:

* определить общие границы и контекст моделируемой предметной области на начальных этапах проектирования системы;
* сформулировать общие требования к функциональному поведению проектируемой системы;
* разработать исходную концептуальную модель системы для ее последующей детализации в форме логических и физических моделей;
* подготовить исходную документацию для взаимодействия разработчиков системы с ее заказчиками и пользователями.

  
Рисунок 1 - Диаграмма вариантов использования базы данных

Концептуальная схема представляет собой описание основных сущностей (таблиц) и связей между ними без учета принятой модели БД и синтаксиса целевой СУБД.

Концептуальная модель включает в себя:

* больница, предоставляет собой множество отделений с кабинетами, в которых врачи выполняют предоставляемые больницей услуги: осмотр, выдача рецепта и т.п. Связь (Кабинет - Отдел), в одном отделе может быть много кабинетов;
* есть множество предоставляемых услуг и поэтому каждой области предоставлено свое отделение, где сотрудники из разных областей не могут работать. Связь (Сотрудник - Отдел).
* тем самым, делаем вывод что в больнице очень много специальностей, и у каждого сотрудника есть своя должность. Связь (Сотрудник - Должность), у одного сотрудника может быть только одна должность;
* чтобы попасть к врачу, необходимо произвести запасись на услугу, т.к. врач не может принять постороннего человека без записи. За каждой услугой закреплен свой врач, а также и отделение. Разные по характеру услуги, предоставляются в разных отделениях. Связь (Запись - Услуга), одна запись может иметь только одну услугу;
* в больнице также присутствуют и платные услуги, после которых клиенту выдаются чеки о том сколько он прошёл услуг и каких именно. Связь (Услуга - Чек), чек может вмещать себя множество услуг, но чек остаётся один;
* также больница имеет в наличии палаты, куда в случаи необходимости кладут клиента для дальнейшего лечения. Есть палаты одноместные, общего назначения или палаты, которые вмещаю в себя один пол людей. Палаты в свою очередь, закреплены за свои отделением. Связь (Палата - Отдел), в одном отделе может быть множество палат;
* во время посещения врача или же услуги, то врач ведет отчет проведенного лечения и общего здоровья клиента. Связь (Запись - Отчет), т.к. за одну запись клиент посещает один раз врача, то и отчет тоже один;
* когда клиент делает запись, у него должна иметься медицинская карта, которую он может завести в больнице, или предоставить медицинскую карту из предыдущей больницы в которой он обслуживался. В медицинскую карту врачи носят историю болезни пациента, аллергии, анализы и т.д. Связь (Клиент – Медицинская карта), у клиента может быть только одна медицинская карта;
  1. **Инфологическое моделирование**

Цель инфологического моделирования – обеспечение наиболее естественных для человека способов сбора и представления той информации, которую предполагается хранить в создаваемой базе данных. Поэтому инфологическую модель данных пытаются строить по аналогии с естественным языком (последний не может быть использован в чистом виде из-за сложности компьютерной обработки текстов и неоднозначности любого естественного языка). Основными конструктивными элементами инфологических моделей являются сущности, связи между ними и их свойства (атрибуты).

Основные понятия:

* сущность – любой различимый объект (объект, который мы можем отличить от другого), информацию о котором необходимо хранить в базе данных. Сущностями могут быть люди, места, самолеты, рейсы, вкус, цвет и т.д. Необходимо различать такие понятия, как тип сущности и экземпляр сущности. Понятие тип сущности относится к набору однородных личностей, предметов, событий или идей, выступающих как целое;
* атрибут – поименованная характеристика сущности. Его наименование должно быть уникальным для конкретного типа сущности, но может быть одинаковым для различного типа сущностей (например, ЦВЕТ может быть определен для многих сущностей: СОБАКА, АВТОМОБИЛЬ, ДЫМ и т.д.). Атрибуты используются для определения того, какая информация должна быть собрана о сущности;
* абсолютное различие между типами сущностей и атрибутами отсутствует. Атрибут является таковым только в связи с типом сущности. В другом контексте атрибут может выступать как самостоятельная сущность. Например, для автомобильного завода цвет – это только атрибут продукта производства, а для лакокрасочной фабрики цвет – тип сущности;
* ключ – минимальный набор атрибутов, по значениям которых можно однозначно найти требуемый экземпляр сущности. Минимальность означает, что исключение из набора любого атрибута не позволяет идентифицировать сущность по оставшимся;
* связь – ассоциирование двух или более сущностей. Если бы назначением базы данных было только хранение отдельных, не связанных между собой данных, то ее структура могла бы быть очень простой. Однако одно из основных требований к организации базы данных – это обеспечение возможности отыскания одних сущностей по значениям других, для чего необходимо установить между ними определенные связи. А так как в реальных базах данных нередко содержатся сотни или даже тысячи сущностей, то теоретически между ними может быть установлено более миллиона связей. Наличие такого множества связей и определяет сложность инфологических моделей.

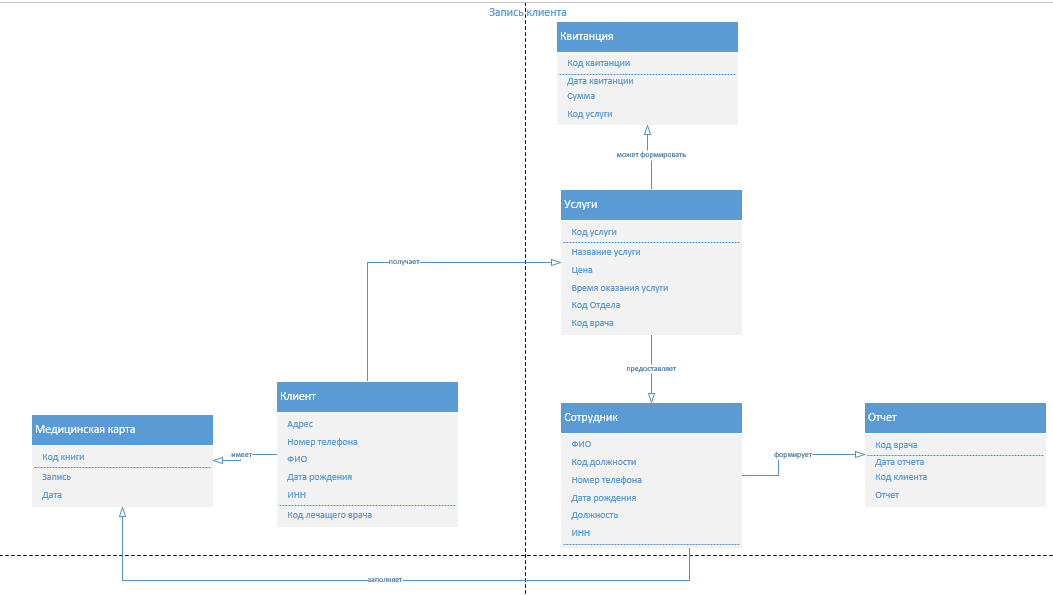


Рисунок 2 – Инфологическая модель БД

* 1. **Построение логической модели**

Логическое проектирование базы данных - процесс создания модели используемой на предприятии информации с учетом выбранной модели организации данных, но независимо от типа целевой СУБД и других физических аспектов реализации. Концептуальная модель данных, созданная на предыдущем этапе, уточняется и преобразуется в логическую модель данных.

Первая нормальная форма:

• запрещает повторяющиеся столбцы (содержащие одинаковую по смыслу информацию);

• запрещает множественные столбцы (содержащие значения типа списка и т.п.);

• требует определить первичный ключ для таблицы, то есть тот столбец или комбинацию столбцов, которые однозначно определяют каждую строку.

Вторая нормальная форма требует, чтобы не ключевые столбцы таблиц завесили от первичного ключа в целом, но не от его части. Маленькая ремарка: если таблица находится в первой нормальной форме и первичный ключ у нее состоит из одного столбца, то она автоматически находится и во второй нормальной форме.

Чтобы таблица находилась в третьей нормальной форме, необходимо, чтобы не ключевые столбцы в ней не зависели от других не ключевых столбцов, а зависели только от первичного ключа. Самая распространенная ситуация в данном контексте - это расчетные столбцы, значения которых можно получить путем каких-либо манипуляций с другими столбцами таблицы. Для приведения таблицы в третью нормальную форму такие столбцы из таблиц надо удалить.

Опираясь на концептуальную модель, созданную на предыдущем этапе, выделим сущности, атрибуты, ключи и связи.

При проектировании базы данных больницы, можно выделить следующие основные сущности:

1. Кабинеты

В данной сущности содержится информация об кабинете: идентификатор кабинета(первичный ключ), для того чтобы их можно было различать; идентификатор работника(внешний ключ), т.к в каждом кабинете должен работать хотя бы один работник; идентификатор отделения(внешний ключ) в котом находиться данный кабинет.

1. Отделения

Для этой сущности предоставлены данные об отделениях: идентификатор (первичный ключ), этаж, название

1. Палаты

Выделены следующие атрибуты данных палаты: идентификатор (первичный ключ), идентификатор отделения (вторичный ключ), сколько мест на данный момент, сколько всего мест в палате, гендер, идентификатор работника (вторичный ключ).

1. Должность

В данной сущности предоставлены данные о существующих должностях: идентификатор (первичный ключ), название.

1. Сотрудники

В данной сущности находиться информация об сотрудниках: идентификатор (первичный ключ), имя, фамилия, отчество, идентификатор должности (внешний ключ), номер мобильного телефона, идентификатор отделения (вторичный ключ), дата рождения, инн.

1. Клиенты

Предоставлены данные, которые позволят идентифицировать каждого клиента: идентификатор (первичный ключ), адрес, номер мобильного телефона, имя, фамилия, отчество(не обязательно), дата рождения, инн, статус, пол.

1. Записи на услугу

Предоставлены данные о записях на услугу: идентификатор (первичный ключ), идентификатор услуги (внешний ключ), идентификатор клиента (внешний ключ), дата начала, идентификатор работника (внешний ключ), идентификатор отделения (внешний ключ).

1. Услуга

В данной сущности выбраны параметры, которые содержат информацию об слуге: идентификатор (первичный ключ), название, продолжительность, цена, идентификатор отделения (внешний ключ), идентификатор работника (внешний ключ), идентификатор кабинета (внешний ключ).

1. Чеки

Для того чтобы можно было узнать прибыль больницы, и для выдачи чеков, были выбраны следующие параметры: идентификатор (первичный ключ), идентификатор клиента (внешний ключ), дата создания чека. В дальнейшем эта сущность будет разбита на две: чек, поля чека. Для более удобного эксплуатирования.

1. Отчет

Выделены атрибуты данных по отчету: идентификатор (первичный ключ), идентификатор работника (внешний ключ), идентификатор клиента (внешний ключ), запись, дата, идентификатор записи (внешний ключ).

1. Медицинская карта

Предоставлены данные о датах заболевания клиента: идентификатор (первичный ключ), дата начала больничного, дата окончания больничного. В дальнейшем для нормализации базы данных, данная таблица была поделена еще на 2 сущности: записи в медицинской карте, записи клиента на услугу.

В результате проектирования базы данных было создано 12 таблиц, имеющие первичный ключ.

Можно выделить три типа связей между таблицами:

• идентифицирующая связь (1:1) — каждой записи одной таблицы соответствует только одна запись другой таблицы;

• один ко многим (1:М) — одной записи главной таблицы могут соответствовать несколько записей подчиненной таблицы;

• многие ко многим (М:М) — одна запись главной таблицы связана с несколькими записями подчиненной таблицы, а одна запись подчиненной таблицы связана с не сколькими записями главной таблицы.

Разработанная логическая модель предоставлена в Приложении А.

1. **Выбор СУБД**

Microsoft SQL Server — система управления реляционными базами данных (СУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для небольших и средних по размеру баз данных, и в последние 5 лет — для крупных баз данных масштаба предприятия, конкурирует с другими СУБД в этом сегменте рынка.

Microsoft SQL Server превосходит MySQL как платформу данных во всех существенных областях, в том числе и по стоимости. Однако стоимость покупки СУБД не столь важна, как ее надежность. Поэтому остановимся на выборе СУБД Microsoft SQL Server.

**2** **Реализация базы данных**

**2.1 Физическое проектирование**

Цель физического проектирования – преобразование логической схемы с учетом синтаксиса, семантики и возможностей выбранной целевой СУБД.

Таблица 1 – Схема БД «Больница»

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Название таблицы** | **Поля таблицы** | **Тип поля** | **Ключ таблицы по полю** | | | **Ограничения** |
| **Первичный ключ** | | **Внешний ключ** |
| Client | Id | int | + | |  | identity |
|  | Adres | Nvarchar(255) |  | | | Not null |
|  | Telephone | Nvarchar(20) |  | | | UNIQUE |
|  | Name | Nvarchar(30) |  | | | Not null |
|  | First\_Name | Nvarchar(30) |  | | | Not null |
|  | Last\_Name | nvarchar (30) |  | | |  |
|  | Birthday | Date |  | | | Not null |
|  | Medical\_Insurance | Int |  | | | Not null |
|  | Status | Nvarchar(40) |  | | |  |
|  | Gender | Bit |  | | | Not null |
| Departament | Id | int | + |  | | identity |
|  | Floor | int |  | | | Not null |
|  | Name | Nvarchar(30) |  | | | Not null |
| Position | Id | int | + |  | | identity |
|  | Name | Nvarchar(30) |  | | | Not null |
| Employee | Id | Int | + |  | | identity |
|  | Name | Nvarchar(25) |  | | | Not null |
|  | First\_Name | Nvarchar(25) |  | | | Not null |
|  | Last\_Name | Nvarchar(25) |  | | |  |
|  | Position\_Id | Int |  | + | | Not null |
|  | Telephone | Nvarchar(20) |  | | | UNIQUE |
|  | Departament\_Id | Int |  | + | | Not null |
|  | Birthday | Date |  | | | Not null |

Продолжение таблицы 1

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Название таблицы** | **Поля таблицы** | **Тип поля** | **Ключ таблицы по полю** | | | | | **Ограничения** |
| **Первичный ключ** | | | **Внешний ключ** | |
|  | Medical\_Insurance | int |  | | | | | UNIQUE,Not null |
| Cabinet | Id | int | + | | |  | | identity |
|  | Employee\_Id | Int |  | | | + | | Not null |
|  | Departament\_Id | int |  | | | + | | Not null |
| Ward | Id | Int | + | | | |  | identity |
|  | Departament\_Id | int |  | | | | + | Not null |
|  | Place | Int |  | | | | | Not null |
|  | Owner\_Place | Int |  | | | | | Not null |
|  | Employee\_Id | Int |  | | + | | | Not null |
|  | Gender | Bit |  | | | | |  |
| Ward\_Client | Id | Int | + | |  | | | identity |
|  | Ward\_Id | Int |  | + | | | | Not null |
|  | Client\_Id | Int |  | | + | | | Not null |
|  | Date\_Invite | Date |  | | | | | Not null |
|  | Date\_Leave | date |  | | | | | Not null |
| Service | Id | int | + | |  | | | identity |
|  | Name | nvarchar(30) |  | | | | | Not null |
|  | Service\_time | Int |  | | | | | Not null |
|  | Price | money |  | | | | | Not null, Check (Price >= 1) |
|  | Departament\_Id | Int |  | | + | | | Not null |
|  | Employee\_Id | Int |  | + | | | | Not null |
|  | Cabinet\_Id | int |  | | + | | | Not null |
| Appointment | Id | int | + | |  | | | identity |
|  | Service\_Id | Int |  | + | | | | Not null |
|  | Client\_Id | Int |  | | + | | | Not null |
|  | Start\_date | datetime |  | | | | | Not null |
|  | Employee\_Id | Int |  | + | | | | Not null |
|  | Departament\_Id | int |  | | + | | | Not null |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Название таблицы** | **Поля таблицы** | **Тип поля** | **Ключ таблицы по полю** | | | **Ограничения** |
| **Первичный ключ** | | **Внешний ключ** |
| Report | Id | int | + | |  | identity |
|  | Client\_Id | int |  | | + | Not null |
|  | Employee\_Id | int |  | | + | Not null |
|  | Recording | nvarchar(1000) |  | |  | Not null |
|  | Date | date |  | | | Not null |
|  | Appointment\_Id | int |  | + | | Not null |
| Receipt | Id | int | + |  | | identity |
|  | Client\_Id | int |  | + | | Not null |
|  | Date\_add | datetime |  | | | Not null |
| Receipt\_Line | Id | int | + |  | | identity |
|  | Receipt\_Id | int |  | + | | Not null |
|  | Service\_Id | int |  | | + | Not null |
|  | Amount | int |  | | | Not null,(Amount >= 1) |
|  | Price | money |  | | | Not null,(Price >= 1) |
| Card\_Line | Id | int | + |  | | identity |
|  | Card\_Id | int |  | + | | Not null |
|  | Recording | Nvarhcar(1000) |  | | | Not null |
|  | Date\_add | datetime |  | | | Not null |
| Medical\_Card | Id | int | + |  | | Not null |
|  | Start\_treatment | date |  | | | Not null |

Продолжение таблицы 1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Название таблицы** | **Поля таблицы** | **Тип поля** | **Ключ таблицы по полю** | | | **Ограничения** |
| **Первичный ключ** | **Внешний ключ** | |
|  | End\_treatment | date |  | | |  |
| Medical\_Card  \_Appointment | Id | int | + |  | | Not null |
|  | Appointment\_Id | int |  | |  | Not null |
|  | Date | date |  | | | Not null |

Продолжение таблицы 1

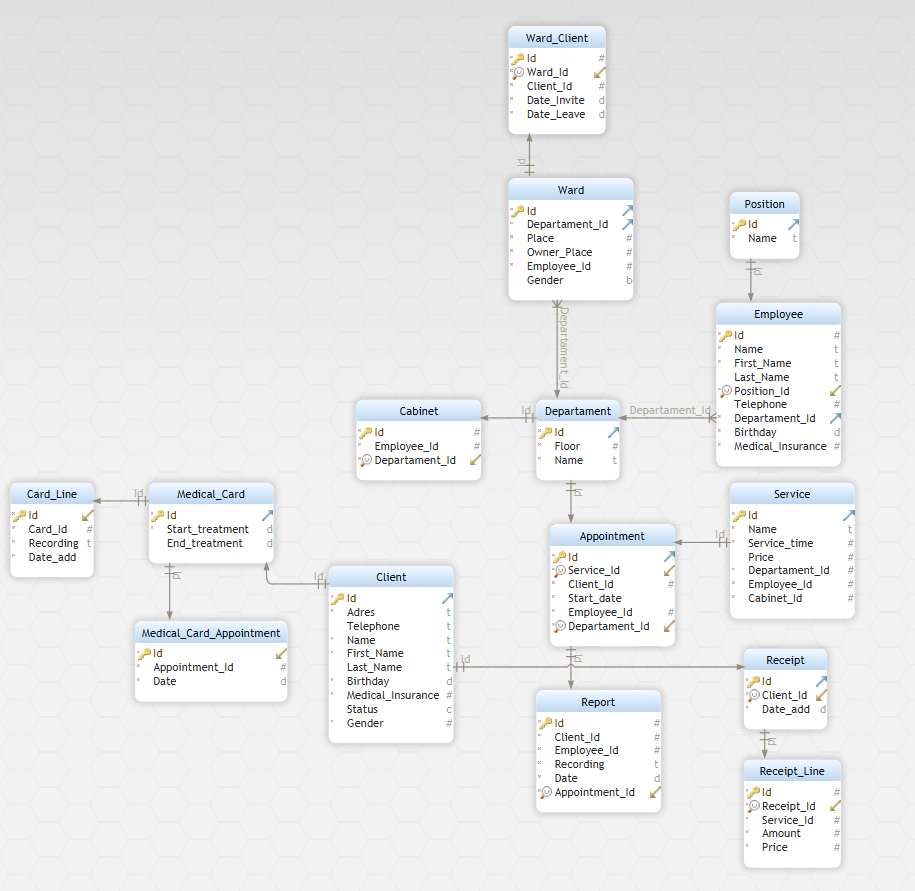


Рисунок 3 – Схема базы данных

**2.2 Написание исходного кода БД**

Создание базы данных:

CREATE DATABASE Hospital

ON PRIMARY --Первичный файл

(

NAME = N'Hospital',

FILENAME = N'C:\DataBases\Hospital.mdf' БД

)

LOG ON

(

NAME = N'Hospital\_log',

FILENAME = N'C:\DataBases\Hospital\_log.ldf'

)

GO

Создание таблицы «Клиенты»:

create table Client

(

Id int identity Primary key,

Adres nvarchar(255) Not null,

Telephone nvarchar(20) UNIQUE,

Name nvarchar(25) Not null,

First\_Name nvarchar(25) Not null,

Last\_Name nvarchar(25),

Birthday Date Not null,

Medical\_Insurance int Not null,

Status nvarchar(40),

Gender bit -- 1 - M 0 - Ж

)

Создание таблицы «Отделение»:

create table Departament

(

Id int identity Primary key,

Floor int Not null,

Name nvarchar(30) Not null

)

Создание таблицы «Должность»:

create table Position

(

Id int identity Primary key,

Name nvarchar(30) Not null

)

Создание таблицы «Работники»:

create table Employee

(

Id int identity Primary key,

Name nvarchar(25) Not null,

First\_Name nvarchar(25) Not null,

Last\_Name nvarchar(25),

Position\_Id int FOREIGN KEY (Position\_Id) References Position(Id) Not null,

Telephone nvarchar(20) UNIQUE,

Departament\_Id int Foreign key (Departament\_Id) References Departament(Id) Not null,

Birthday Date Not null,

Medical\_Insurance int Not null

)

Создание таблицы «Кабинеты»:

create table Cabinet

(

Id int identity Primary key,

Employee\_Id int Foreign key (Employee\_Id) References Employee(Id) Not null,

Departament\_Id int Foreign key (Departament\_Id) References Departament(Id) Not null

)

Создание таблицы «Палаты»:

create table Ward

(

Id int identity Primary key,

Departament\_Id int Not null Foreign key (Departament\_Id) References Departament(Id),

Place int Not null,

Owner\_Place int Not null,

Employee\_Id int Foreign key (Employee\_Id) References Employee(Id) Not null,

Gender bit

)

Создание таблицы «Клиенты в палате»:

create table Ward\_Client

(

Id int identity Primary key,

Ward\_Id int Not null Foreign key (Ward\_Id) references Ward(Id),

Client\_Id int Not null Foreign key (Client\_Id) references Client(Id),

Date\_Invite date Not null,

Date\_Leave date Not null

)

Создание таблицы «Услуги»:

create table Service

(

Id int identity Primary key,

Name nvarchar(30) Not null,

Service\_time int Not null, --время указанное в минутах

Price MONEY Not null Check (Price >= 1),

Departament\_Id int Foreign key (Departament\_Id) References Departament(Id) Not null,

Employee\_Id int Foreign key (Employee\_Id) References Employee(Id) Not null,

Cabinet\_Id int Foreign key (Cabinet\_Id) References Cabinet(Id) Not null

)

Создание таблицы «Записи»:

create table Appointment

(

Id int identity Primary key,

Service\_Id int Foreign key (Service\_Id) References Service(Id) Not null,

Client\_Id int Foreign key (Client\_Id) References Client(Id) Not null,

Start\_date datetime Not null,

Employee\_Id int Foreign key (Employee\_Id) References Employee(Id) Not null,

Departament\_Id int Foreign key (Departament\_Id) References Departament(Id) Not null

)

Создание таблицы «Отчет»:

create table Report

(

Id int identity Primary key,

Client\_Id int Foreign key (Client\_Id) References Client(Id) Not null,

Employee\_Id int Foreign key (Employee\_Id) References Employee(Id) Not null,

Recording nvarchar(1000) Not null,

Date datetime Not null,

Appointment\_Id int Foreign key (Appointment\_Id) References Appointment(Id) Not null

)

Создание таблицы «Чек»:

create table Receipt

(

Id int identity Primary key,

Client\_Id int Foreign key (Client\_Id) References Client(Id) Not null,

Date\_add datetime

)

Создание таблицы «Поля чека»:

create table Receipt\_Line

(

Id int identity Primary key,

Receipt\_Id int Foreign key (Receipt\_Id) References Receipt(Id) Not null,

Service\_Id int Foreign key (Service\_Id) References Service(Id) Not null,

Amount int Not null Check (Amount >= 1),

Price Money Not null Check (Price >= 1)

)

Создание таблицы «Медицинская карта»:

create table Medical\_Card

(

Id int Foreign key (Id) References Client(Id) Not null,

Start\_treatment datetime,

End\_treatment datetime,

)

Создание таблицы «Запись в медицинской карте»:

create table Card\_Line

(

Id int identity Primary key,

Card\_Id int Foreign key (Card\_Id) References Client(Id) Not null,

Recording nvarchar(1000) Not null,

Date\_add datetime Not null

)

Создание таблицы «Записи клиента»:

create table Medical\_Card\_Appointment

(

Id int Foreign key (Id) References Client(Id) Not null,

Appointment\_Id int Foreign key (Appointment\_Id) References Appointment(Id) Not null,

Date datetime Not null

)

**2****.3 Написание запросов, функций и хранимых процедур**

**2.3.1 Общие сведения**

Запросы служат для извлечения данных из таблиц и предоставления их пользователю в удобном виде. С помощью запросов выполняются такие операции как отбор данных, их сортировка и фильтрация. С помощью запросов можно выполнять преобразования данных по заданному алгоритму, создавать новые таблицы, выполнять автоматическое наполнение таблиц данными, импортированными из других источников, выполнять простейшие вычисления в таблицах и многое другое. Особенность запросов состоит в том, что они черпают данные из базовых таблиц и создают на их основе временную результирующую таблицу. Если хотят подчеркнуть факт «временности» этой таблицы, то ее еще называют моментальным снимком [3]. В данной курсовой работе предлагается создать несколько запросов: запросы на выборку, запросы с использованием диалогового окна для ввода условия, запросы с постоянно заданным условием.

**2.3.2 Хранимые процедуры**

Создание хранимой процедура для добавления «Клиента»:

drop procedure Add\_Client

create procedure Add\_Client(@Adres nvarchar(255),@Telephone nvarchar(20),@Name nvarchar(25),@First\_Name nvarchar(25)

,@Last\_Name nvarchar(25),@Birthday Date,@Medical\_Insurance int,@Status nvarchar(40),@Gender bit)

As

Begin

insert into Client(Adres,Telephone,Name,First\_Name,Last\_Name,Birthday,Medical\_Insurance,Status,Gender) values

(@Adres,@Telephone,@Name,@First\_Name,@Last\_Name,@Birthday,@Medical\_Insurance,@Status,@Gender)

End

Go

Создание хранимой процедура для добавления «Отделения»:

drop procedure Add\_Departament

create procedure Add\_Departament(@Floor int,@Name nvarchar(30))

As

Begin

Insert into Departament(Floor,Name) values

(@Floor,@Name)

End

Go

Создание хранимой процедура для добавления «Работника»:

drop procedure Add\_Employee

create procedure Add\_Employee(@Name nvarchar(25),@First\_Name nvarchar(25),@Last\_Name nvarchar(25)

,@Position\_Id int,@Telephone nvarchar(20),@Departament\_Id int,@Birthday Date,@Medical\_Insurance int)

As

Begin

Insert into Employee(Name,First\_Name,Last\_Name,Position\_Id,Telephone,Departament\_Id,Birthday,Medical\_Insurance) values

(@Name,@First\_Name,@Last\_Name,@Position\_Id,@Telephone,@Departament\_Id,@Birthday,@Medical\_Insurance)

End

Go

Создание хранимой процедура для добавления «Должности»:

drop procedure Add\_Position

create procedure Add\_Position(@Name nvarchar(30))

As

Begin

Insert into Position(Name) values

(@Name)

End

Go

Создание хранимой процедура для добавления «Кабинета»:

drop procedure Add\_Cabinet

create procedure Add\_Cabinet(@Employee\_Id int,@Departament\_Id int)

As

Begin

Insert into Cabinet(Employee\_Id,Departament\_Id) values

(@Employee\_Id,@Departament\_Id)

End

Go

Создание хранимой процедура для добавления «Палаты»:

drop procedure Add\_Ward

create procedure Add\_Ward(@Departament\_Id int,@Place int,@Owner\_Place int,@Employee\_Id int,@Gender bit)

As

Begin

Insert into Ward(Departament\_Id,Place,Owner\_Place,Employee\_Id,Gender) values

(@Departament\_Id,@Place,@Owner\_Place,@Employee\_Id,@Gender)

End

Go

Создание хранимой процедура для добавления «Клиента в палате»:

drop procedure Add\_Ward\_Client

create procedure Add\_Ward\_Client(@Ward\_Id int,@Client\_Id int,@Date\_Invite date,@Date\_Leave date)

As

Begin

Insert into Ward\_Client(Ward\_Id,Client\_Id,Date\_Invite,Date\_Leave) values

(@Ward\_Id,@Client\_Id,@Date\_Invite,@Date\_Leave)

End

Go

Создание хранимой процедура для добавления «Услуги»:

drop prucedure Add\_Service

create procedure Add\_Service(@Name nvarchar(30),@Service\_time int,@Price MONEY,@Departament\_Id int,@Employee\_Id int,

@Cabinet\_Id int)

As

Begin

Insert into Service(Name,Service\_time,Price,Departament\_Id,Employee\_Id,Cabinet\_Id) values

(@Name,@Service\_time,@Price,@Departament\_Id,@Employee\_Id,@Cabinet\_Id)

End

Go

Создание хранимой процедура для добавления «Записи»:

drop procedure Add\_Appointment

create procedure Add\_Appointment(@Service\_Id int,@Client\_Id int,@Start\_date datetime,@Employee\_Id int,@Departament\_Id int)

As

Begin

Insert into Appointment(Service\_Id,Client\_Id,Start\_date,Employee\_Id,Departament\_Id) values

(@Service\_Id,@Client\_Id,@Start\_date,@Employee\_Id,@Departament\_Id)

End

Go

Создание хранимой процедура для добавления «Отчета»:

drop procedure Add\_Report

create procedure Add\_Report(@Client\_Id int,@Employee\_Id int,@Recording nvarchar(1000),@Appointment\_Id int)

As

Begin

Declare @Date datetime

Select @Date = SYSDATETIME()

Insert into Report(Client\_Id,Employee\_Id,Recording,Date,Appointment\_Id) values

(@Client\_Id,@Employee\_Id,@Recording,@Date,@Appointment\_Id)

Insert into Card\_Line(Card\_Id,Recording,Date\_add) values

(@Client\_Id,@Recording,@Date)

End

Go

Создание хранимой процедура для добавления «Чека»:

drop procedure Add\_Receipt

create procedure Add\_Receipt(@Client\_Id int,@Date\_add datetime)

As

Begin

insert into Receipt(Client\_Id,Date\_add) values

(@Client\_Id,@Date\_add)

End

Go

Создание хранимой процедура для добавления «Полей чека»:

drop procedure Add\_Receipt\_Line

create procedure Add\_Receipt\_Line(@Receipt\_Id int,@Service\_Id int,@Amount int)

As

Begin

Declare @Price money

Select @Price = Price \* @Amount from Service where Id = @Service\_Id

insert into Receipt\_Line(Receipt\_Id,Service\_Id,Amount,Price) values

(@Receipt\_Id,@Service\_Id,@Amount,@Price)

End

Go

Создание хранимой процедура для добавления «Медицинской карты»:

drop procedure Add\_Medical\_Card

create procedure Add\_Medical\_Card(@Id int,@Start\_treatment datetime,@End\_treatment datetime)

As

Begin

insert into Medical\_Card(Id,Start\_treatment,End\_treatment) values

(@Id,@Start\_treatment,@End\_treatment)

End

Go

Создание хранимой процедура для добавления «Записей в мед. карту»:

drop procedure Add\_Card\_Line

create procedure Add\_Card\_Line(@Card\_Id int,@Recording nvarchar(1000))

As

Begin

Declare @Date datetime

Select @Date = SYSDATETIME()

insert into Card\_Line(Card\_Id,Recording,Date\_add) values

(@Card\_Id,@Recording,@Date)

End

Go

Создание хранимой процедура для добавления «Записей клиента»:

drop procedure Add\_Medical\_Card\_Appointment

create procedure Add\_Medical\_Card\_Appointment(@Id int,@Appointment\_Id int,@Date datetime)

As

Begin

insert into Medical\_Card\_Appointment(Id,Appointment\_Id,Date) values

(@Id,@Appointment\_Id,@Date)

End

Go

**2.3.3 Запросы на заполнение таблиц**

Заполнение таблицы «Клиента»:

--Для клиента (Адрес, Номер телефона, Имя, Фамилия, Отчество, Дата рождения, ИНН )

exec Add\_Client 'Пл.Свободы 15','89610549314','Денис','Гурьянов','Денисович',N'12-06-2003',05120485,null,1

exec Add\_Client 'цл.максима горького 43','89657492756','Владислав','Иванов','Владович',N'12-12-2005',00536234,null,1

exec Add\_Client 'ул.покровская дом 5 кв 5','89075648506','Егор','Смирнов','Егорович',N'12-11-2006',12344563,null,1

exec Add\_Client '10-й Сквозной переулок 2','89776057864','Маргарита','Киселева','Маргаритовна',N'12-10-2006',05122445,null,0

exec Add\_Client 'улица 20-го Партсъезда 3','896754295642','Данияр','Кузнецов','Даниярович',N'12-09-2005',02420485,null,1

exec Add\_Client '5-й Березовый переулок 3','89078562492','Алексей','Попов','Алексеевич',N'12-08-2001',07924515,null,1

exec Add\_Client 'улица Вавилова 1','89061758231','Дмитрий','Васильев','Дмитриевич',N'12-07-2001',90657421,null,1

exec Add\_Client 'улица Абая 1','8906546714','Александр','Петров','Александрович',N'18-01-2002',00067582,null,1

exec Add\_Client 'улица Гагарина 2','89099967581','Андрей ','Соколов','Андреевич',N'22-05-2008',11857300,null,1

exec Add\_Client 'Абазанская улица 9','89608461842','Артем','Михайлов','Артемович',N'17-01-2006',46571900,null,1

exec Add\_Client '8-й Вязовский переулок 7','89610500653','Богдан','Новиков','Богданович',N'25-02-2001',09876543,null,1

exec Add\_Client 'переулок Иванова 5','89610549874','Тамара','Соловьева','Тамаровна',N'27-03-2002',98674927,null,0

exec Add\_Client 'улица Жана Жореса 10','89610564735','Анатолий','Федоров','Анатольевич',N'02-04-1999',08563856,null,1

exec Add\_Client 'Кавказская улица 16','89665748296','Борис','Морозов','Борисович',N'14-07-2005',06417598,null,1

exec Add\_Client 'Набережная улица 24','89067584127','Василий','Волков','Васильевич',N'11-06-2003',52389043,null,1

exec Add\_Client 'Пл.Свободы 12','89609657421','Раиса','Андреева','Раисовна',N'16-10-2003',74531675,null,0

exec Add\_Client 'улица Радищева 2','89610565778','Ульяна','Степанова','Ульяновна',N'27-12-2001',16534949,null,0

exec Add\_Client 'Удмуртская улица 1','89610987609','Даниил','Алексеев','Даниилович',N'29-11-2002',23796034,null,1

exec Add\_Client 'Фабричная улица 2','89610560042','Глеб','Павлов','Глебович',N'10-01-2001',64318903,null,0

exec Add\_Client 'Шатурский переулок 5','89610768592','Нина','Козлова','Нановна',N'09-02-2000',05235789,null,0

Заполнение таблицы «Отделения»:

--Для отделения(Этаж, Название)

exec Add\_Departament 1,'Первое'

exec Add\_Departament 2,'Второе'

exec Add\_Departament 3,'Третье'

exec Add\_Departament 4,'Четвертое'

exec Add\_Departament 2,'Пятое'

exec Add\_Departament 3,'Шестое'

exec Add\_Departament 4,'Седьмое'

exec Add\_Departament 2,'Возьмое'

exec Add\_Departament 2,'Голубиное'

exec Add\_Departament 4,'Тортова'

Заполнение таблицы «Должности»:

--Для должности(Название/Имя)

exec Add\_Position 'Главный врач'

exec Add\_Position 'Директор больницы'

exec Add\_Position 'Акушер-гинеколог'

exec Add\_Position 'Гематолог'

exec Add\_Position 'Гериатр'

exec Add\_Position 'Детский онколог-гематолог'

exec Add\_Position 'Диабетолог'

exec Add\_Position 'Медицинский микробиолог'

exec Add\_Position 'Терапевт'

exec Add\_Position 'Токсиколог'

exec Add\_Position 'Уролог'

Заполнение таблицы «Работника»:

--Для работника(Имя, Фамилия, Отчество, Йд должности, Номер телефона, Йд отделения, Дата рождения, ИНН)

exec Add\_Employee 'Денис','Гурьянов','Денисович',1,'89610549314',1,N'12-06-2003',05120485

exec Add\_Employee 'Владислав','Иванов','Владович',2,'89061758233',2,N'12-12-2005',00536234

exec Add\_Employee 'Егор','Смирнов','Егорович',3,'89610500652',3,N'12-11-2006',12344563

exec Add\_Employee 'Маргарита','Киселева','Маргаритовна',4,'89610500651',4,N'12-10-2006',05122445

exec Add\_Employee 'Данияр','Кузнецов','Даниярович',5,'89610549875',5,N'12-08-2001',07924515

exec Add\_Employee 'Алексей','Попов','Алексеевич',6,'89609657426',6,N'12-07-2001',90657421

exec Add\_Employee 'Дмитрий','Васильев','Дмитриевич',7,'89609657427',7,N'18-01-2002',00067582

exec Add\_Employee 'Андрей ','Соколов','Андреевич',8,'89061758238',8,N'22-05-2008',11857300

exec Add\_Employee 'Артем','Михайлов','Артемович',8,'89078562499',9,N'17-01-2006',46571900

exec Add\_Employee 'Тамара','Соловьева','Тамаровна',9,'89078562490',10,N'25-02-2001',09876543

exec Add\_Employee 'Нина','Козлова','Нановна',10,'89610549311',2,N'02-04-1999',08563856

exec Add\_Employee 'Даниил','Алексеев','Даниилович',11,'89610565772',1,N'14-07-2005',06417598

exec Add\_Employee 'Тамара','Соловьева','Тамаровна',3,'89665748293',4,N'11-06-2003',52389043

exec Add\_Employee 'Ульяна','Степанова','Ульяновна',5,'8906546714',6,N'16-10-2003',74531675

exec Add\_Employee 'Борис','Морозов','Борисович',4,'89610560045',4,N'27-12-2001',16534949

exec Add\_Employee 'Анатолий','Федоров','Анатольевич',7,'89610768596',5,N'29-11-2002',23796034

Заполнение таблицы «Палаты»:

--Для палата (Йд отделения, Количество свободных мест, Всего мест, Гендер, Йд работника)

exec Add\_Ward 1,15,15,1,0

exec Add\_Ward 1,1,1,2,1

exec Add\_Ward 2,2,2,3,null

exec Add\_Ward 3,10,10,4,1

exec Add\_Ward 6,12,12,5,0

exec Add\_Ward 9,9,9,6,null

exec Add\_Ward 5,5,5,7,1

Заполнение таблицы «Клиента в палате»:

--Для клиентов в палате

exec Add\_Ward\_Client 13,1,N'12-06-2022',N'13-06-2022'

exec Add\_Ward\_Client 13,12,N'14-06-2022',N'20-06-2022'

exec Add\_Ward\_Client 13,3,N'15-06-2022',N'16-06-2022'

exec Add\_Ward\_Client 13,4,N'16-06-2022',N'17-07-2022'

exec Add\_Ward\_Client 13,5,N'17-06-2022',N'18-06-2022'

exec Add\_Ward\_Client 13,11,N'18-06-2022',N'25-06-2022'

exec Add\_Ward\_Client 13,7,N'19-06-2022',N'20-06-2022'

exec Add\_Ward\_Client 13,10,N'20-06-2022',N'21-08-2022'

exec Add\_Ward\_Client 13,9,N'21-06-2022',N'22-06-2022'

Заполнение таблицы «Услуги»:

--Для услуги(Имя, Время проведения в мин, Цена, Йд отделение, Йд работника, Кабинет)

exec Add\_Service 'Обследование',200,5600,1,1,1

exec Add\_Service 'Обследование',200,2000,2,2,1

exec Add\_Service 'Операция',200,10000,3,3,1

exec Add\_Service 'Обследование',200,8500,4,4,1

exec Add\_Service 'Обследование',200,500,5,5,1

exec Add\_Service 'Обследование',200,1000,6,6,1

Заполнение таблицы «Кабинета»:

--Для кабинета (Йд работника, Йд отделения)

exec Add\_Cabinet 1,1

exec Add\_Cabinet 2,2

exec Add\_Cabinet 3,3

exec Add\_Cabinet 4,4

exec Add\_Cabinet 5,5

exec Add\_Cabinet 6,6

exec Add\_Cabinet 7,7

exec Add\_Cabinet 8,8

Заполнение таблицы «Записи»:

--Для записи на услугу 21-02-2012 6:10:00 PM

exec Add\_Appointment 7,1,'21-05-2022 12:45:09 PM',1,1

exec Add\_Appointment 8,2,'22-05-2022 12:45:09 PM',1,1

exec Add\_Appointment 9,3,'23-05-2022 12:45:09 PM',1,1

exec Add\_Appointment 11,4,'24-05-2022 12:45:09 PM',1,1

exec Add\_Appointment 12,5,'25-05-2022 12:45:09 PM',1,1

exec Add\_Appointment 8,2,'26-05-2022 12:45:09 PM',1,1

Заполнение таблицы «Отчета»:

exec Add\_Report 1,1,'Эта была суровая ночь, клиент бился в агонии ночью за курсовой работой. Он всю ночь бубнил себе под нос – “Я больше не хочу испытывать такого ужаса, который я испытал за 2 месяца”. Буду надеяться, что все уладидтся',6

Заполнение таблицы «Чека»:

--Для чека

exec Add\_Receipt 1,'21-05-2022 12:45:09 PM'

exec Add\_Receipt 2,'22-05-2022 12:45:09 PM'

exec Add\_Receipt 3,'23-05-2022 12:45:09 PM'

exec Add\_Receipt 4,'24-05-2022 12:45:09 PM'

Заполнение таблицы «Полей чека»:

--Для полей чека (Йд чека, Йд услуги, Колличество)

exec Add\_Receipt\_Line 1,10,2

exec Add\_Receipt\_Line 1,8,1

exec Add\_Receipt\_Line 1,11,2

**2.3.4 Триггеры**

Триггер на корректное внесение данных в «Клиенты в палате»:

drop trigger client\_in\_ward

create trigger client\_in\_ward on Ward\_Client INSTEAD OF INSERT

As

Begin

if exists(Select Client\_Id, w.Gender from inserted i inner join

Ward w on i.Ward\_Id = w.Id inner join

Client c on i.Client\_Id = c.Id where (w.Gender = c.Gender) or (w.Gender is null)) -- Проверка подходит ли палата клиенту по гендору(полу)

Begin

if (Select w.Place from inserted i inner join

Ward w on i.Ward\_Id = w.Id) > 0 -- Проверка есть ли места в палате

Begin

if not exists(Select TOP(1) i.Client\_Id from inserted i inner join

Ward\_Client wc on wc.Client\_Id = i.Client\_Id where i.Client\_Id = wc.Client\_Id) -- Проверка на есть ли клиент уже в какой то палате

Begin

Insert into Ward\_Client(Ward\_Id,Client\_Id,Date\_Invite)

Select Ward\_Id,Client\_Id,Date\_Invite from inserted

Update Ward SET Place = (Place - 1) where Id = (Select i.Ward\_Id from inserted i)

End

Else

Begin

Print 'Вы пытаетесь записать килента который уже находиться в палате, в другую палату'

ROLLBACK TRANSACTION

End

End

End

Else -- Блок проверок какая именно ошибка в добавлении

Begin

if exists(Select i.Client\_Id from inserted i inner join

Ward\_Client wc on wc.Client\_Id = i.Client\_Id ) --Проверка на есть ли клиент уже в какой то палате

Begin

Print 'Вы пытаетесь записать килента который уже находиться в палате, в другую палату'

ROLLBACK TRANSACTION

End

Else

Begin

if not exists(Select w.Gender from inserted i inner join

Ward w on i.Ward\_Id = w.Id inner join

Client c on i.Client\_Id = c.Id where w.Gender = c.Gender) -- Проверка подходит ли палата клиенту по гендору(полу)

Begin

Print 'Вы пытаетесь добавить клиента палату не по гендору'

ROLLBACK TRANSACTION

End

Else

Begin

if (Select w.Place from inserted i inner join

Ward w on i.Ward\_Id = w.Id) = 0 -- Проверка есть ли места в палате

Begin

Print 'Вы пытаетесь добавить клиента палату в которой нету мест'

ROLLBACK TRANSACTION

End

End

End

End

Go

**2.3.5 Хранимые процедуры с запросами**

Запрос на количество мест в палатах общего назначения:

drop procedure Mesta\_for\_All

create procedure Mesta\_for\_All

As

Begin

Select SUM(Place) from Ward where Gender is null

End

Go

Запрос на количество мест в палатах для женщин:

drop procedure Mesta\_for\_W

create procedure Mesta\_for\_W

As

Begin

Select SUM(Place) from Ward where Gender = 0

End

Go

Запрос на количество мест в палатах для мужчин:

drop procedure Mesta\_for\_M

create procedure Mesta\_for\_M

As

Begin

Select SUM(Place) from Ward where Gender = 1

End

Go

Запрос на вывод информации по среднему времени препровождения в палате:

drop procedure Sred

create procedure Sred

As

Begin

Select AVG(DATEDIFF(DAY, Date\_Invite, Date\_Leave)) from Ward\_Client

End

Go

Запрос на срок лечения определенного клиента:

drop procedure Time\_of\_treatment

create procedure Time\_of\_treatment(@Client\_Id int)

As

Begin

Select DATEDIFF(DAY, Start\_treatment, End\_treatment) from Medical\_Card where Id = @Client\_Id

End

Go

Запрос на вывод информации по количеству прооперированных пациентов (с осложнениями и умерших):

drop procedure Operated\_patients

create procedure Operated\_patients

As

Begin

Select SUM(c.Id),'Осложнения' = (Select SUM(c.Id) where Status = 'Осложнения'),'Умерший' = (Select SUM(c.Id) where Status = 'Умерший' ) from Appointment a inner join

Client c on a.Client\_Id = c.Id inner join

Medical\_Card mc on mc.Id = c.Id where (Service\_Id = 3) and (mc.End\_treatment is null)

Go

End

**Заключение**

Разработанная база данных «Больницы» позволяет быстро и эффективно работать с данным данной предметной области. Данная БД является учебной и не охватывает всю бизнес-логику. Однако является прототипом, демонстрирующим работу в данной отрасли. Данная БД может быть расширена для автоматизации нерассмотренных в рамках данной курсовой работы концепций в предметной области «Больницы». Были изучены и проработаны следующие пункты:

* был проведен анализ поставленной задачи и изучена предметная область;
* был рассмотрен и усвоен предложенный теоретический материал, что позволило более качественно и быстро справиться с поставленными целями и задачами;
* были получены практические навыки по использованию системы управления базы данных MS SQL Server;
* таблицы были заполнены необходимыми данными.

Главной целью разработанной системы это автоматизация рабочего места работников, с целью уменьшить долю ручного труда и количества бумажных документов.

**Приложение А**

