Частное образовательное учреждение высшего образования

«Казанский инновационный университет имени В. Г. Тимирясова (ИЭУП)»

Факультет менеджмента и инженерного бизнеса

Кафедра информационных технологий и техносферной безопасности

КУРСОВОЙ ПРОЕКТ

по дисциплине «Базы данных»

на тему: «База данных «Платная поликлиника»

Направление подготовки: 09.03.03 «Прикладная информатика»

Выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

обучающийся гр. 1001у

заочной формы обучения

Султаняров В.А.

Руководитель: \_\_\_\_\_\_\_\_\_\_\_

канд. экон. наук, доцент каф. ИТиТБ

Григорян К.А.

Казань, 2021 г

**АННОТАЦИЯ**

на курсовой проект

по дисциплине «Базы данных»

|  |
| --- |
| Ф.И.О. студента: Султаняров Владислав Алексеевич |
| Student’s Name: Sultanyarov Vladislav Alekseevich |
| Направление: 09.03.03 «Прикладная информатика» |
| № группы: 1001у |

|  |
| --- |
| Название курсового проекта: |
| База данных «Платная поликлиника» |
| Paper Title: |
| Database "Paid polyclinic" |

Аннотация:

|  |
| --- |
| В курсовой работе проводится исследование и проектирование базы данных |
| для предметной области “Платная поликлиника” |
|  |
|  |
|  |
|  |
|  |

Summary:

|  |  |
| --- | --- |
| In the course work, research and design | |
| of a database for the subject area "Paid Polyclinic" | |
|  | |
|  | |
|  | |
|  | |
| Руководитель: | Перевод проверил: |
| «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г. | «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2020 г. |

[ВВЕДЕНИЕ 3](#_Toc88393143)

[1. ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 4](#_Toc88393144)

[2. КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 5](#_Toc88393145)

[2.1. Определение сущностей 5](#_Toc88393146)

[2.2. Определение атрибутов 6](#_Toc88393147)

[3. ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 7](#_Toc88393148)

[3.1. Нормализация отношений 7](#_Toc88393149)

[3.2. Классификация связей 8](#_Toc88393150)

[4. ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 10](#_Toc88393151)

[5. РАБОТА С БАЗОЙ ДАННЫХ 17](#_Toc88393152)

[5.1. Создание запросов 17](#_Toc88393153)

[5.2. Создание представлений 25](#_Toc88393154)

[5.3. Создание функций 27](#_Toc88393155)

[5.4. Создание триггеров 28](#_Toc88393156)

[ЗАКЛЮЧЕНИЕ 30](#_Toc88393157)

[СПИСОК ЛИТЕРАТУРЫ 31](#_Toc88393158)

# ВВЕДЕНИЕ

На сегодняшний день трудно представить сферу жизни, в которую не проникли информационные технологии. Так и в медицине – для обеспечения более высокого качества обслуживания пациентов врачам нужно использовать различное ПО, которое позволяет сэкономить драгоценное время для работы с пациентом. Врачам больше не требуется вручную заполнять огромное количество справок, отчетов и других документов, необходимых в их работе [3].

Поликлиника — это организация, которая предназначена для контроля и диагностики состояния здоровья граждан. Следовательно, поликлиника работает с очень большим объемом информации, как о сотрудниках, так и о пациентах. Врачам необходимо всегда следить за данными о своих пациентах, о курсе лечения больных. А руководству и бухгалтерии необходимо быть в курсе событий о своих сотрудниках. Для этого нужна общая база данных, включающая всю необходимую информацию.

База данных «Платная поликлиника» включает в себя данные о врачах, пациентах и обращениях, которые необходимые для работы поликлиники. База данных позволяет осуществлять добавление, изменение, поиск и удаление данных, а также просматривать эти данные.

Актуальность данной темы в том, что в наш век информационных технологий, стало реально все документы преобразовывать в электронный вид и регистратура в считанные минуты может найти сведения о принятых пациентах, обращениях пациентов к врачам, врачах [5].

Цель работы: собрать материал и разработать базу данных для работы платной поликлиники.

# ИССЛЕДОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ

Предметной областью называется часть реальной системы, представляющая интерес для данного исследования. При проектировании автоматизированных информационных систем предметная область отображается моделями данных нескольких уровней. Число уровней зависти от сложности решаемых задач, но в любом случае включает концептуальный и логический уровни.

В данной курсовой работе предметной областью является работа платной поликлиники, задачей которой выступает предоставление услуг пациентам.

Поликлиника - многопрофильное или специализированное лечебно-профилактическое учреждение для оказания амбулаторной медицинской помощи больным на приёме и на дому. Клиент – потребитель услуг поликлиники. Врач – специалист, предоставляющий услуги.

# КОНЦЕПТУАЛЬНОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

Концептуальное (инфологическое) проектирование - построение семантической модели предметной области, то есть информационной модели наиболее высокого уровня абстракции. Такая модель создаётся без ориентации на какую-либо конкретную СУБД и модель данных [8]. Термины «семантическая модель», «концептуальная модель» и «инфологическая модель» являются синонимами. Кроме того, в этом контексте равноправно могут использоваться слова «модель базы данных» и «модель предметной области» (например, «концептуальная модель базы данных» и «концептуальная модель предметной области»), поскольку такая модель является как образом реальности, так и образом проектируемой базы данных для этой реальности.

Конкретный вид и содержание концептуальной модели базы данных определяется выбранным для этого формальным аппаратом. Обычно используются графические нотации, подобные ER-диаграммам [10].

Эта база данных может быть использована для накопления информации для статистической отчетности, поэтому можно сказать, что это система информационного обеспечения многопользовательская. Основная ее функция отслеживание записей на прием пациентов поликлиники.

В данном разделе формирование перечня сущностей и атрибутов базы данных “Платная поликлиника” осуществляется в два этапа

* Определение сущностей;
* Определение атрибутов.

# Определение сущностей

Сущность в базе данных – это любой объект в базе данных, который можно выделить исходя из сути предметной области, для которой разрабатывается эта база данных. Разработчик базы данных должен уметь правильно определять сущности.

База данных “Платная поликлиника” включает в себя следующие сущности:

1. “Клиент” – люди, которые когда-либо обращались в поликлинику;
2. “Специальность врача” – перечень медицинский специальностей, по которым предоставляются услуги в поликлинике;
3. “Услуга” – список услуг, оказываемых в поликлинике;
4. “Должность” – должности, по которым производится трудоустройство сотрудников поликлиники;
5. “Сотрудник” – перечень сотрудников, работающих в поликлинике;
6. “Запись” – записи клиентов на приемы;
7. “Медицинская карта” – медицинские карты всех клиентов поликлиники.

# Определение атрибутов

Атрибут представляет собой свойство сущности.

Выделим атрибуты для объектов базы данных “Платная поликлиника”.

1. Клиент (id, фио, дата\_рождения, домашний\_адрес, номер\_телефона);
2. Специальность врача (id, название\_специальности);
3. Услуга (id, id\_специальности\_врача, название\_услуги, базовая\_цена, минимальное\_время\_услуги);
4. Должность (id, название\_должности, зарплата);
5. Сотрудник (id, фио, id\_специальности\_врача, дата\_рождения, опыт\_работы, id\_должности);
6. Запись (id, id\_клиента, id\_сотрудника, id\_услуги, время\_приема, кабинет, примечания\_к\_приему);
7. Медицинская карта (id, id\_клиента, id\_записи, эпикриз);

# ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

**Логическое проектирование базы данных** – это процесс создания модели используемой на предприятии информации на основе выбранной модели организации данных, но без учета типа целевой СУБД и других физических аспектов реализации.

Его цель состоит в создании логической модели данных для исследуемой части предприятия. Концептуальная модель данных, созданная на предыдущем этапе, уточняется и преобразуется в логическую модель данных. Логическая модель данных учитывает особенности выбранной модели организации данных в целевой СУБД (например, реляционная модель) [9].

Перед тем, как построить логическую модель, необходимо нормализовать отношения, классифицировать связи и выбрать ключи.

# Нормализация отношений

Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение [1].

Процесс преобразования отношений базы данных к виду, отвечающему нормальным формам, называется нормализацией. Нормализация предназначена для приведения структуры БД к виду, обеспечивающему минимальную логическую избыточность, и не имеет целью уменьшение или увеличение производительности работы или же уменьшение или увеличение физического объёма базы данных [6].

База данных “Платная поликлиника” находится в третьей нормальной форме (3НФ), так как отвечает требованиям первой, второй и третьей нормальных форм.

Первая нормальная форма. В таблице отсутствуют повторяющиеся строки; в каждой ячейке таблицы хранится только одно типовое не составное значение, а также отсутствуют списки и массивы в любой форме.

Вторая нормальна форма. База находится в первой нормальной форме; все таблицы имеют уникальный ключ. Например:

1. В таблице “Клиенты” ключом выступает столбец “id”;
2. В таблице “Специальность врача” ключом выступает столбец “id”;

Третья нормальная форма. База данных находится во второй нормальной форме, в таблицах отсутствуют транзитивные зависимости, то есть, каждый неключевой столбец зависит только от первичного ключа.

# Классификация связей

В базе данных “Платная поликлиника” определены следующие отношения между таблицами:

Таблица 1 – Отношения между таблицами

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Родительская таблица | Дочерняя таблица | Ключи | | Вид связи |
| 1 | Специальности | Услуги | id | id\_специальности\_врача | 1: ∞ |
| 2 | Специальности | Сотрудники | id | id\_специальности\_врача | 1: ∞ |
| 3 | Должности | Сотрудники | id | id\_должности | 1: ∞ |
| 4 | Клиенты | Записи | id | id\_клиента | 1: ∞ |
| 5 | Сотрудники | Записи | id | id\_сотрудника | 1: ∞ |
| 6 | Услуги | Записи | id | id\_записи | 1: ∞ |
| 7 | Клиенты | Медицинская карта | id | id\_клиента | 1: ∞ |
| 8 | Медицинская карта | Записи | id | id\_записи | 1: ∞ |

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 1 - ER-диаграмма

# ФИЗИЧЕСКОЕ ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

В физической модели описываются типы, идентификаторы и разрядность полей. Физическая модель данных отражает физическое размещение данных на машинных носителях, то есть, какой файл, какие объекты, с какими атрибутами содержит и каковы типы этих атрибутов [4].

Таблица 2 – физическая модель таблицы “Клиенты”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код клиента | Счетчик | Целое | Уникальный идентификатор клиента |
| 2 | ФИО | Текстовый | Переменная длина | ФИО клиента |
| 3 | Дата рождения | Дата | Календарная дата | Дата рождения клиента |
| 4 | Домашний адрес | Текстовый | Переменная длина | Место жительства клиента |
| 5 | Мобильный телефон | Текстовый | 10 | Мобильный телефон клиента |

Таблица 3 – физическая модель таблицы “Специальности врачей”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код специальности | Счетчик | Целое | Уникальный идентификатор специальности |
| 2 | Название | Текстовый | Переменная длина | Название специальности |

Таблица 3 – физическая модель таблицы “Услуги”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код услуги | Счетчик | Целое | Уникальный идентификатор услуги |
| 2 | Код специальности | Целое | Целое | Идентификатор специальности, к которой относится услуга |
| 3 | Название | Текстовый | Переменная длина | Название услуги |
| 4 | Цена | Денежный |  | Цена за услугу |
| 5 | Время | Целое |  | Время выполнения услуги (мин) |

Таблица 4 – физическая модель таблицы “Должности”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код должности | Счетчик | Целое | Уникальный идентификатор должности |
| 2 | Название | Текстовый | Переменная длина | Название должности |
| 3 | Зарплата | Денежный |  | Зарплата |

Таблица 5 – физическая модель таблицы “Сотрудники”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код сотрудника | Счетчик | Целое | Уникальный идентификатор сотрудника |
| 2 | ФИО | Текстовый | Переменная длина | ФИО сотрудника |
| 3 | Код специальности | Целое |  | Специальности сотрудника (только у врачей) |
| 4 | Дата рождения | Дата | Календарная дата | Дата рождения сотрудника |
| 5 | Опыт | Целое |  | Стаж сотрудника |
| 6 | Код должности | Целое |  | Код должности сотрудника |

Таблица 6 – физическая модель таблицы “Записи”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код записи | Счетчик | Целое | Уникальный идентификатор записи |
| 2 | Код клиента | Целое |  | Код клиента, идущего на прием |
| 3 | Код сотрудника | Целое |  | Код сотрудника, оказывающего услугу на приеме |
| 4 | Код специальности | Целое |  | Код специальности, по которой оказывается услуга |
| 5 | Время | Дата и время с учетом пояса |  | Время приема |
| 6 | Кабинет | Текстовое | 5 | Кабинет |
| 7 | Заметка | Текстовое | Переменная длина | Примечания к приему |

Таблица 7 – физическая модель таблицы “Медицинские карты”

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Поле | Тип | Размер | Описание |
| 1 | Код медицинской карты | Счетчик | Целое | Уникальный идентификатор медицинской карты |
| 2 | Код клиента | Целое |  | Идентификатор клиента |
| 3 | Код записи | Целое |  | Код записи |
| 4 | Эпикриз | Текстовое | Переменная длина | Заключение врача |

# РАБОТА С БАЗОЙ ДАННЫХ

# Создание запросов

Запрос для получения всех врачей и услуги, которые они предоставляют:

-- Получить всех врачей и услуги, которые они предоставляют

SELECT e.id,

e.fio,

service\_name

FROM service s

INNER JOIN employee e ON s.medic\_specialty\_id = e.medic\_specialty\_id;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 2 - Результат выполнения запроса для получения всех врачей и услуг, которые они предоставляют

Запрос для получения всех сотрудников с сортировкой по зарплате:

-- Получить всех сотрудников с сортировкой по зарплате

SELECT e.fio,

p.salary

FROM employee e

INNER JOIN position p ON p.id = e.position\_id

ORDER BY p.salary DESC;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 3 - Результат выполнения запроса на получение всех сотрудников с сортировкой по зарплате

Запрос для получения списка всех услуг в поликлинике:

-- Получить список всех услуг в поликлинике

SELECT s.service\_name,

e.fio

FROM service s

LEFT JOIN employee e ON s.medic\_specialty\_id = e.medic\_specialty\_id;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 4 - Результат выполнения запроса для получения списка всех услуг в поликлинике

Запрос для получения всех должностей клиники:

-- Получить все должности клиники

SELECT p.name,

p.salary,

e.fio

FROM position p

RIGHT JOIN employee e ON p.id = e.position\_id;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 5 - Результат выполнения запроса для получения всех сотрудников и их должностей

Запрос для получения суммы зарплат всех врачей:

-- Получить сумму зарплату всех врачей

SELECT SUM(p.salary)

FROM position p

INNER JOIN employee e ON p.id = e.position\_id

WHERE e.medic\_specialty\_id IS NOT NULL

GROUP BY CASE WHEN e.medic\_specialty\_id IS NULL THEN 'NULL' ELSE 'NOT NULL' END;

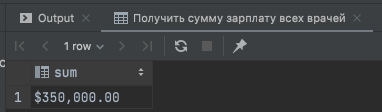


Рисунок 6 - Результат выполнения запроса на получение суммы зарплат всех врачей

Запрос для получения списка всех специальностей и всех услуг:

-- Получить список всех специальностей и всех услуг

SELECT ms.name

FROM medic\_specialty ms

UNION ALL

SELECT s.service\_name

FROM service s;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 7 - Результат выполнения запроса для получения всех специальностей и всех услуг

Запрос для получения всех врачей, на которых назначено больше одной записи:

-- Получить всех врачей, на которых назначено больше одной записи

SELECT e.fio

FROM employee e

WHERE e.id IN ((SELECT a.employee\_id

FROM appointment a

GROUP BY a.employee\_id

HAVING COUNT(a.employee\_id) > 1));

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 8 - Результат выполнения запроса для получения всех врачей, на которых назначено больше одной записи

Запрос для получения медицинской карты Бойцовой Любомилы Игоревны:

-- Получить мед.карту Бойцовой Любомилы Игоревны

SELECT c.fio,

em.fio,

s.service\_name

FROM medical\_history m

INNER JOIN client c ON c.id = m.client\_id

INNER JOIN appointment e ON e.id = m.appointment\_id

INNER JOIN employee em ON em.id = e.employee\_id

INNER JOIN service s ON s.id = e.service\_id

WHERE m.client\_id = (

SELECT id

FROM client

WHERE fio = 'Бойцова Любомила Игоревна');

Изображение выглядит как текст, снимок экрана, монитор, экран

Автоматически созданное описание

Рисунок 9 - Результат выполнения запроса для получения медицинской карты Бойцовой Любомилы Игоревны

Запрос для получения списка всех клиентов, у которых был хотя бы один прием:

-- Получить список всех клиентов, у которых был хотя бы один прием

SELECT c.fio

FROM client c

INNER JOIN appointment a ON c.id = a.client\_id

GROUP BY c.fio;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 10 - Результат выполнения запроса для получения всех клиентов, у которых был хотя бы один прием

Запрос для получения списка всех сотрудников с их должностями:

-- Получить список всех сотрудников с должностями

SELECT e.fio,

p.name,

p.salary,

ms.name,

e.experience

FROM employee e

INNER JOIN position p ON p.id = e.position\_id

LEFT JOIN medic\_specialty ms ON ms.id = e.medic\_specialty\_id

ORDER BY e.fio;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 11 - Результат выполнения запроса для получения всех сотрудников с их должностями и зарплатами

Запрос для получения суммы ежемесячных выплат по зарплатам:

-- Получить сумму ежемесячных выплат по зарплатам

SELECT SUM(p.salary)

FROM employee e

INNER JOIN position p ON p.id = e.position\_id;

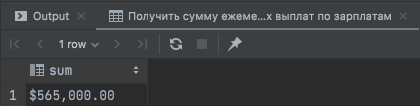


Рисунок 12 - Результат выполнения запроса для получения суммы ежемесячных выплат по зарплатам

Запрос для получения среднего стажа всех сотрудников:

-- Получить средний стаж всех сотруднков

SELECT AVG(e.experience)

FROM employee e;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 13 - Результат выполнения запроса для получения среднего стажа сотрудников поликлиники

Запрос для получения представленных в клинике специальности:

-- Получить представленные в клинике специальности

SELECT ms.name

FROM medic\_specialty ms

INNER JOIN employee e ON ms.id = e.medic\_specialty\_id;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 14 - Результат выполнения запроса для получения всех представленных специальностей

Запрос для получения средней стоимости услуг по каждому клиенту:

-- Получить среднюю стоимость услуг по каждому клиенту

SELECT c.fio,

AVG(s.fix\_price::numeric)

FROM medical\_history mh

INNER JOIN appointment a ON a.id = mh.appointment\_id

INNER JOIN service s ON s.id = a.service\_id

INNER JOIN client c on c.id = mh.client\_id

GROUP BY mh.client\_id, c.fio;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 15 - Результат выполнения запроса для получения средней стоимости услуг по каждому клиенту

Запрос для получения списка услуг, для которых необходимо нанять специалистов:

-- Список услуг, для которых необходимо нанять специалистов

SELECT s.service\_name

FROM service s

LEFT JOIN employee e ON s.medic\_specialty\_id = e.medic\_specialty\_id

WHERE e.medic\_specialty\_id IS NULL;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 16 - Результат выполнения запроса для получения списка услуг, для которых необходимо нанять специалистов

# Создание представлений

Запрос на создание представления с информацией о необходимых сотрудниках:

-- Создаем представление с информацией о необходимых сотрудниках

CREATE VIEW NEEDED\_STAFF AS

SELECT s.service\_name

FROM service s

LEFT JOIN employee e ON s.medic\_specialty\_id = e.medic\_specialty\_id

WHERE e.medic\_specialty\_id IS NULL;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 17 - Результат обращения к представлению с информацией о необходимых сотрудниках

Запрос на создание представления с прайс-листом:

-- Создаем представление с прайс-листом

CREATE VIEW PRICE\_LIST AS

SELECT s.service\_name, s.fix\_price, s.min\_time

FROM service s;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 18 - Результат обращения к представлению с прайс-листом

Запрос на создание представления со списком врачей

-- Создаем представление со списком врачей

CREATE VIEW DOC\_LIST AS

SELECT e.fio, ms.name, e.experience

FROM employee e

INNER JOIN medic\_specialty ms on ms.id = e.medic\_specialty\_id;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 19 - Результат обращения к представлению со списком врачей

# Создание функций

Запрос на создание функции для получения сотрудников с указанной или выше зарплатой:

-- Создаем функцию для получения сотрудников с указанной или выше зарплатой [2]

CREATE OR REPLACE FUNCTION find\_employees\_with\_indicated\_salary(employeeSalary NUMERIC)

RETURNS SETOF INT

LANGUAGE plpgsql AS

$$

DECLARE

employee\_id int;

BEGIN

FOR employee\_id IN SELECT e.id

FROM employee e

INNER JOIN position p ON p.id = e.position\_id

WHERE p.salary::numeric >= employeeSalary

LOOP

RETURN NEXT employee\_id;

END LOOP;

END

$$;

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 20 - Результат выполнения функции для получения сотрудников с указанной или выше зарплатой

# Создание триггеров

Запрос на создание триггера и подключение его к таблице с записями для создания записей в медицинской карте:

-- Создадим триггер для создания записи в мед.карте во время создания приема

CREATE OR REPLACE FUNCTION insert\_medical\_history() RETURNS TRIGGER

LANGUAGE plpgsql AS

$$

BEGIN

INSERT INTO medical\_history(client\_id, appointment\_id) VALUES (NEW.client\_id, NEW.id);

RETURN NEW;

END;

$$;

-- Вешаем триггер на таблицу приемов

CREATE TRIGGER update\_medical\_history

AFTER INSERT

ON appointment

FOR EACH ROW

EXECUTE PROCEDURE insert\_medical\_history();

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 21 - Таблица медицинских карт до выполнения триггера

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 22 - Таблица медицинских карт после выполнения триггера

# ЗАКЛЮЧЕНИЕ

Современные программные системы становятся сложнее, чтобы обеспечить возможность решения глобальных задач, например, таких, как создание единой системы управления предприятием. При разработке таких систем важно хорошо представлять современные подходы, существующие в этой области, и основные сложности этого процесса.

Потребность в сложных программных системах растет. По мере того, как увеличивается производительность и падает цена вычислительной техники, появляются возможности выполнить автоматизацию все более сложных процессов. Основная ценность проектирования при создании сложных информационных систем состоит в том, что оно позволяет свести к минимуму трудоемкую рутинную работу и сосредоточиться на решении творческих задач [7].

В результате выполнения данной курсовой работы были достигнут цели, поставленные в начале работы. Была разработана структура базы данных, созданы запросы, представления, процедуры и триггеры.

# СПИСОК ЛИТЕРАТУРЫ

1. Глушаков С.В., Ломотько Д.В. Базы данных: Учебный курс. - Харьков: Фолио; Ростов н/Д: Феникс; Киев: Абрис, 2000. - 504 с.
2. Документация к PostgreSQL 14.1 [Электронный ресурс]. URL: https://postgrespro.ru/docs/postgresql/14/
3. Информационные технологии. УМК. М.Р. Павлова. - СПб.: изд-во СЗТУ, 2008.
4. Карпова Т.С. Базы данных: модели, разработка, реализация. - СПб: Питер, 2002.
5. Корнеев В.В. и др. Базы данных. Интеллектуальная обработка информации // М.: Нолидж, 2000. - 352 с.
6. Крёнке Д. Теория и практика построения баз данных. - СПб: Питер, 2003.
7. Мишенин А.И. Теория экономических информационных систем - М.: Финансы и статистика, 1999. - 168 с.
8. Реляционная СУБД [Электронный ресурс]. URL: ru.wikipedia.org/wiki/Реляционная\_СУБД
9. Сигнор Р., Стегман М.О. Использование ОDBС для доступа к базам данных - М.: БИНОМ, 1995. - 384 с.
10. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы данных. Учебник для ВУЗов /под ред. проф.А.Д. Хомоненко // СПб.: КОРОНАпринт, 2000. - 416 с.