Министерство транспорта Российской Федерации

Федеральное государственное автономное образовательное  
учреждение высшего образования  
«РУТ (МИИТ)»

Институт транспортной техники и систем управления

Кафедра «Управление и защита информации»

КУРСОВОЙ ПРОЕКТ

по дисциплине

**«Основы построения защищенных баз данных»**

**на тему  
«Поликлиника»**

Выполнил: ст. гр. ТКИ-441

Куминов В. П.

Проверил: доц., к.т.н.

Васильева М. А.

Москва 2023

Оглавление

[ЦЕЛЬ КУРСОВОГО ПРОЕКТА\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3](#_Toc135498336)

[ЗАДАНИЕ НА КУРСОВОВОЙ ПРОЕКТ\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3](#_Toc135498337)

[ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ\_\_\_\_\_\_\_\_\_\_\_4](#_Toc135498338)

[1. Инфологическое проектирование 4](#_Toc135498339)

[1.1 Анализ предметной области 4](#_Toc135498340)

[1.2 Анализ информационных задач и круга пользователей системы\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5](#_Toc135498341)

[2. Логическое проектирование реляционной БД 5](#_Toc135498342)

[2.1 Составление реляционных отношений 5](#_Toc135498343)

[2.2 Нормализация полученных отношений 11](#_Toc135498344)

[3. Физическое проектирование БД 12](#_Toc135498345)

[3.1 Разработка скриптов на создание базы данных и таблиц 12](#_Toc135498346)

[3.2 Разработка скриптов на добавление данных в таблицы 15](#_Toc135498347)

[3.3 Разработка необходимых запросов: 17](#_Toc135498348)

[3.4 Разработка необходимых запросов, представлений (view), процедур, функций и триггеров: 19](#_Toc135498349)

[Список литературы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_22](#_Toc135498350)

# ЦЕЛЬ КУРСОВОГО ПРОЕКТА

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) Система управления базами данных Postgres.

# ЗАДАНИЕ НА КУРСОВОВОЙ ПРОЕКТ

В данном курсовом проекте ставится задача разработать РБД в СУБД DBeaver по заданной теме «Поликлиника». Проектирование РБД проводится с помощью метода «Сущность-связь». Проверка построенной модели РБД осуществляется с помощью метода нормализации отношений.

**Описание предметной области.**

База данных создаётся для информационного обслуживания регистрационного отдела поликлиники. БД должна содержать информацию о врачах, ведущих прием, расписании приема, и пациентах, льготах пациентах, кабинете врача.

**Готовые запросы:**

1. Выдавать сводную информацию обо всех врачах поликлиники.
2. Выдавать сводную информацию о пациентах.
3. Выдавать информацию о записи пациента к врачу.
4. Выдавать информацию о приеме врачей на указанную дату.
5. Выдавать информацию о пациентах, имеющих льготы на приобретение лекарств.

# ПРОЕКТИРОВАНИЕ РЕЛЯЦИОННОЙ БАЗЫ ДАННЫХ

## Инфологическое проектирование

### Анализ предметной области

База данных создаётся для информационного обслуживания некой поликлиники.

Выделим базовые сущности этой предметной области:

* Doctor
* Patient
* Schedule

ER-диаграмма БД, приведена на (Рисунке 1). На данной ER-диаграмме присутствуют связь один - ко - многим.

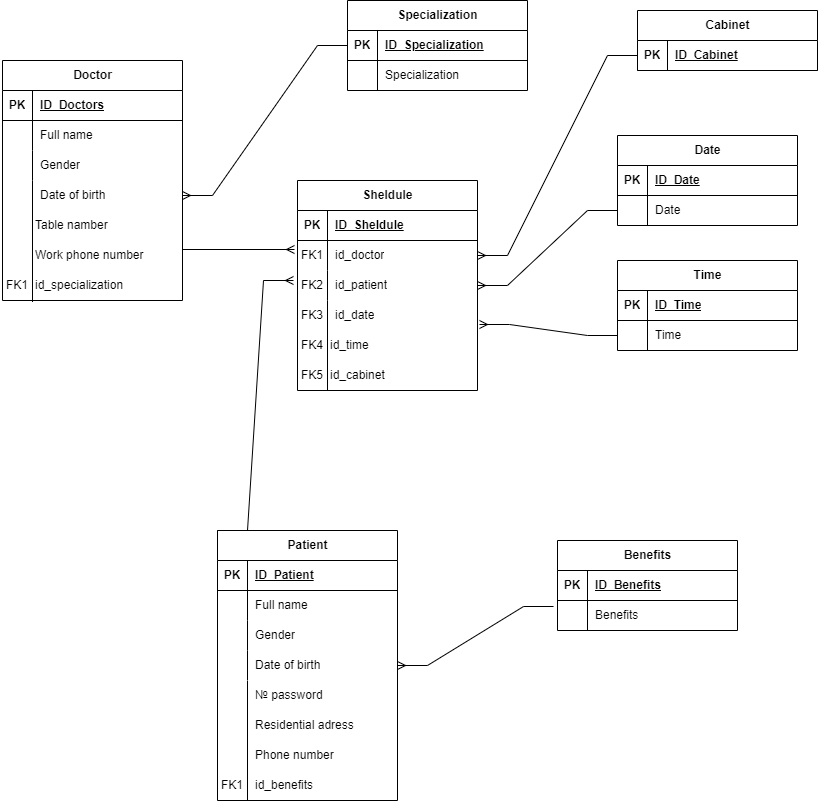


Рисунок 1 – ER-диаграмма

### Анализ информационных задач и круга пользователей системы

Система создаётся для обслуживания следующих групп пользователей:

* Врач
* Пациент

Определим границы информационной поддержки пользователей:

**1) Функциональные возможности:**

* ведение БД (запись, чтение, модификация, удаление);
* обеспечение логической непротиворечивости БД;
* реализация наиболее часто встречающихся запросов в готовом виде.

**2) Готовые запросы:**

* Выдавать сводную информацию обо всех врачах поликлиники.
* Выдавать сводную информацию о пациентах.
* Выдавать информацию о записи пациента к врачу.
* Выдавать информацию о приеме врачей на указанную дату.
* Выдавать информацию о пациентах, имеющих льготы на приобретение лекарств.

## Логическое проектирование реляционной БД

### Составление реляционных отношений

Каждое реляционное отношение соответствует одной сущности (объекту ПрО), и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть). В том случае, если базовое отношение не имеет потенциальных ключей, вводится *суррогатный первичный ключ*, который не несёт смысловой нагрузки и служит только для идентификации записей.

Были созданы следующие сущности:

1. Таблица "Specialization" - содержит информацию о специализации врачей.
2. Таблица "Doctor" - содержит информацию о врачах, такую как ID\_Doctors, полное имя, пол, дата рождения, табельный номер, рабочий телефон и идентификатор специализации.
3. Таблица "Cabinet" - содержит номера кабинетов, где проходят приемы пациентов.
4. Таблица "Patient" - содержит информацию о пациентах, такую как ID\_Patient, полное имя, пол, дата рождения, номер паспорта, адрес проживания, телефон и идентификатор наличия льгот.
5. Таблица "Benefits" - содержит информацию о льготах, доступных пациентам
6. Таблица "Date" - содержит даты приемов пациентов.
7. Таблица "Time" - содержит время приема пациентов.
8. Таблица "Schedule" - содержит расписание приемов пациентов, включая идентификатор врача, пациента, дату и время приема и номер кабинета.

**Таблица 1 – Схема отношения** **Доктор** (Doctors)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Код доктора | ID\_Doctors | Целое число | - | Первичный ключ, генерируемый автоматически |
| ФИО | Full name | Текст | - | Обязательное поле |
| Пол | Gender | Символ | 3 | Обязательное поле |
| Дата рождения | Date of birth | Дата | - | Обязательное поле |
| Табельный номер | Table number | Символ | 6 | Обязательное поле |
| Рабочий телефон | Work phone number | Символ | 5 | Обязательное поле |
| Код специализации | id\_specialization | Целое число | - | Внешний ключ, ссылается на таблицу "Specialization" |

Таблица 2 – Схема отношения Специализация (Specialization)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| ID специализации | ID\_Specialization | Целое число | - | Первичный ключ, генерируемый автоматически |
| Специализация | Specialization | Текст | - | Обязательное поле |

Таблица 3 – Схема отношения Пациент (Patient)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Код пациента | ID\_Patient | Целое число | - | Первичный ключ, генерируемый автоматически |
| ФИО | Full name | Текст | - | Обязательное поле |
| Пол | Gender | Текст | - | Обязательное поле |
| Дата рождения | Date of birth | Дата | - | Обязательное поле |
| Номер паспорта | № passport | Текст | 10 | Обязательное поле |
| Адрес проживания | Residential address | Текст | 250 | Обязательное поле |
| Номер телефона | Phone number | Текст | 11 | Обязательное поле |
| Код льготы | id\_benefits | Целое число | - | Внешний ключ, ссылается на таблицу "Benefits" |

Таблица 4 – Схема отношения Льгота (Benefits)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | ID\_Benefits | Integer | - | Первичный ключ, генерируемый автоматически |
| Льгота | Benefits | Boolean | - | Обязательное поле |

Таблица 5 – Схема отношения Кабинет (Cabinet)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | ID\_Cabinet | SERIAL | - | Первичный ключ, автоинкремент |

Таблица 6 – Схема отношения Дата (Date)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | ID\_Date | integer |  | Первичный ключ, генерируемый автоматически |
| Дата | Date | date |  | Обязательное поле |

Таблица 7 – Схема отношения Время (Time)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор | ID\_Time | integer |  | Первичный ключ, генерируемый автоматически |
| Время приема | Time of admission | time |  | Обязательное поле |

Таблица 8– Схема отношения Расписание (Schedule)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Длина** | **Примечания** |
| Идентификатор записи | ID\_Schedule | integer | - | Первичный ключ, генерируемый автоматически |
| Идентификатор доктора | id\_doctor | integer | - | Внешний ключ, ссылающийся на таблицу "Doctor" |
| Идентификатор пациента | id\_patient | integer | - | Внешний ключ, ссылающийся на таблицу "Patient" |
| Идентификатор даты приёма | id\_date of admission | integer | - | Внешний ключ, ссылающийся на таблицу "Date" |
| Идентификатор времени приёма | id\_time of admission | integer | - | Внешний ключ, ссылающийся на таблицу "Time" |
| Идентификатор кабинета | id\_cabinet | integer | - | Внешний ключ, ссылающийся на таблицу "Cabinet" |

### Нормализация полученных отношений

**1НФ.** Для приведения таблиц к 1НФ требуется составить таблицы (один атрибут – один столбец) и разбить сложные атрибуты на простые, а многозначные атрибуты вынести в отдельные отношения.  
В наших таблицах все атрибуты простые.

**2НФ.** Все отношения находятся во 2НФ, так как для каждого из них введен уникальный ключ.

**3НФ**. Все неключевые атрибуты зависят только от первичного ключа и не зависят друг от друга. Поэтому спроектированная **база данных находится в третьей нормальной форме**.

Созданную схему Вы можете наблюдать на рисунке 2.

Изображение выглядит как текст, снимок экрана, диаграмма, линия

Автоматически созданное описание

Рисунок 2 – ER-диаграмма в DBeaver

## Физическое проектирование БД

### Разработка скриптов на создание базы данных и таблиц

БД «Поликлиника» проектируется в среде Система управления базами данных PostgresSQL.

*Создание базы данных Поликлиника:*

USE public;

Создание таблицы Специализация:

create TABLE "Specialization" (

"ID\_Specialization" integer not null generated always as identity PRIMARY key,

"Specialization" text not null

);

*Создание таблицы Доктор:*

CREATE TABLE "Doctor" (

"ID\_Doctors" integer not null generated always as identity PRIMARY KEY,

"Full name" text not null,

"Gender" char(3) not null,

"Date of birth" date not null,

"Table number" char(6) not null,

"Work phone number" char(5) not null,

"id\_specialization" integer not null,

FOREIGN KEY (id\_specialization) REFERENCES "Specialization" ("ID\_Specialization")

);

*Создание таблицы Кабинет:*

create TABLE "Cabinet" (

"ID\_Cabinet" SERIAL PRIMARY KEY

);

*Создание таблицы Дата:*

CREATE TABLE "Date" (

"ID\_Date" integer not null generated always as identity PRIMARY key,

"Date" date not null);

*Создание таблицы Время:*

CREATE TABLE "Time" (

"ID\_Time" integer not null generated always as identity PRIMARY key,

"Time of admission" time not null

);

*Создание таблицы Льгота:*

CREATE TABLE "Benefits" (

"ID\_Benefits" integer not null generated always as identity PRIMARY key,

"Benefits" boolean

);

*Создание таблицы Пациент:*

CREATE TABLE "Patient" (

"ID\_Patient" integer not null generated always as identity PRIMARY KEY,

"Full name" text,

"Gender" text,

"Date of birth" date not null,

"№ passport" char(10) not null,

"Residential address" varchar(250) not null,

"Phone number" char(11) not null,

"id\_benefits" integer not null,

FOREIGN KEY (id\_benefits) REFERENCES "Benefits" ("ID\_Benefits")

*Создание таблицы Расписание:*

CREATE TABLE "Schedule" (

"ID\_Schedule" integer not null generated always as identity PRIMARY key,

"id\_doctor" integer not null,

"id\_patient" integer not null,

"id\_date of admission" integer not null,

"id\_time of admission" integer not null,

"id\_cabinet" integer not null,

FOREIGN KEY (id\_doctor) REFERENCES "Doctor" ("ID\_Doctors"),

FOREIGN KEY (id\_patient) REFERENCES "Patient" ("ID\_Patient"),

FOREIGN KEY ("id\_date of admission") REFERENCES "Date" ("ID\_Date"),

FOREIGN KEY ("id\_time of admission") REFERENCES "Time" ("ID\_Time"),

FOREIGN KEY (id\_cabinet) REFERENCES "Cabinet" ("ID\_Cabinet")

);

### 3.2 Разработка скриптов на добавление данных в таблицы

*Заполнение таблицы Специализация:*

insert into "Specialization" ("Specialization") values

('Терапевт'),

('Офтальмолог'),

('Отоларинголог'),

('Гинеколог'),

('Стоматолог'),

('Врач общей практики'),

('Хирург'),

('Гастроэнтеролог'),

('Проктолог'),

*Заполнение таблицы Доктор:*

insert into "Doctor" ("Full name", "id\_specialization", "Gender", "Date of birth", "Table number", "Work phone number") values

('Петраков Андрей Васильевич',1, 'МУЖ', '12.01.1987', 152041, 24325),

('Никифоров Никита Евгеньевич',2, 'МУЖ', '22.11.1958', 324512, 87657),

('Киракосян Анна Феликсовна',3, 'ЖЕН', '14.09.1974', 191631, 32567)

*Заполнение таблицы Кабинет:*

INSERT into "Cabinet" VALUES

(101),

(102),

(103),

(104),

*Заполнение таблицы Дата:*

insert into "Date" ("Date") values

('05.05.2023'),

('06.05.2023'),

('07.05.2023'),

('08.05.2023'),

('09.05.2023'),

*Заполнение таблицы Время:*

insert into "Time" ("Time of admission") values

('07:00'),

('07:15'),

('07:30'),

('07:45'),

Заполнение таблицы Льгота:

insert into "Benefits" ("Benefits") values

('No'),

('Yes')

*Заполнение таблицы Пациент:*

insert into "Patient" ("Full name", "Gender", "Date of birth", "№ passport" , "Residential address","Phone number",id\_benefits) values

('Смирнова Елена Петровна', 'ЖЕН', '12.10.1985', 5314682487, 'г. Москва, ул. Ленина, дом 25, кв. 18', 89991234567, 1),

('Иванов Игорь Владимирович', 'МУЖ', '23.05.1978', 1234567890, 'г. Москва, ул. Красная, дом 5, кв. 47', 89054561234, 1),

*Заполнение таблицы Расписание:*

insert into "Schedule" ("id\_doctor", "id\_patient", "id\_date of admission", "id\_time of admission", "id\_cabinet") values

(1, 2, 1, 1, 101),

(3, 4, 2, 4, 205),

(5, 6, 3, 7, 315),

(2, 3, 4, 13, 110),

### Разработка необходимых запросов:

*Выдавать сводную информацию обо всех врачах поликлиники.*

SELECT "Doctor"."ID\_Doctors", "Doctor"."Full name", "Specialization"."Specialization"

FROM "Doctor"

JOIN "Specialization" ON "Doctor"."id\_specialization" = "Specialization"."ID\_Specialization";

*Выдавать сводную информацию о пациентах.*

SELECT \*

FROM "Patient" AS a;

*Выдавать информацию о записи пациента к врачу.*

SELECT

"Schedule"."ID\_Schedule",

"Doctor"."Full name" as "Doctor name",

"Patient"."Full name" as "Patient name",

"Date"."Date",

"Time"."Time of admission",

"Cabinet"

FROM

"Schedule"

JOIN "Doctor" ON "Schedule"."id\_doctor" = "Doctor"."ID\_Doctors"

JOIN "Patient" ON "Schedule"."id\_patient" = "Patient"."ID\_Patient"

JOIN "Date" ON "Schedule"."id\_date of admission" = "Date"."ID\_Date"

JOIN "Time" ON "Schedule"."id\_time of admission" = "Time"."ID\_Time"

JOIN "Cabinet" ON "Schedule"."id\_cabinet" = "Cabinet"."ID\_Cabinet"

*Выдавать информацию о приеме врачей на указанную дату.*

SELECT "Doctor"."Full name" AS "Doctor's name", "Patient"."Full name" AS "Patient's name",

"Date"."Date" AS "Date of admission", "Time"."Time of admission" AS "Time of admission",

"Cabinet" AS "Number cabinet"

FROM "Schedule"

JOIN "Doctor" ON "Doctor"."ID\_Doctors" = "Schedule"."id\_doctor"

JOIN "Patient" ON "Patient"."ID\_Patient" = "Schedule"."id\_patient"

JOIN "Date" ON "Date"."ID\_Date" = "Schedule"."id\_date of admission"

JOIN "Time" ON "Time"."ID\_Time" = "Schedule"."id\_time of admission"

JOIN "Cabinet" ON "Cabinet"."ID\_Cabinet" = "Schedule"."id\_cabinet"

WHERE "Date"."Date" = '2023-05-08';

*Выдавать информацию о пациентах, имеющих льготы на приобретение лекарств.*

SELECT \*

FROM "Patient"

JOIN "Benefits" ON "Patient"."id\_benefits" = "Benefits"."ID\_Benefits"

WHERE "Benefits"."Benefits" = true;

### Разработка необходимых запросов, представлений (view), процедур, функций и триггеров:

*Пример представления view:*

--Самый пожилой доктор

CREATE VIEW "OldestDoctor" AS

SELECT "Full name", "Date of birth"

FROM "Doctor"

WHERE "Date of birth" = (

SELECT MIN("Date of birth")

FROM "Doctor"

);

--Отображение представления view

SELECT \*

FROM "OldestDoctor";

*Пример процедуры:*

--Получить список имен врачей, связанных с определенной специализацией, на основе переданного идентификатора специализации

CREATE OR REPLACE PROCEDURE get\_doctors\_by\_specialization(

p\_specialization\_id INTEGER

)

LANGUAGE plpgsql

AS $$

BEGIN

PERFORM "Full name"

FROM "Doctor"

WHERE "id\_specialization" = p\_specialization\_id;

END;

$$;

-- вызов процедуры

CALL get\_doctors\_by\_specialization(1);

*Пример функции:*

--Считаем количество врачей определенной специальности

CREATE OR REPLACE FUNCTION count\_doctors\_by\_specialization(p\_specialization\_id INTEGER)

RETURNS INTEGER AS $$

DECLARE

result INTEGER;

BEGIN

SELECT COUNT(\*)

INTO result

FROM "Doctor"

WHERE "id\_specialization" = p\_specialization\_id;

RETURN result;

END;

$$ LANGUAGE plpgsql;

--Вызов функции

SELECT count\_doctors\_by\_specialization(1);

-- Создание триггера

CREATE OR REPLACE FUNCTION count\_doctors\_trigger\_func()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'INSERT' OR TG\_OP = 'UPDATE' OR TG\_OP = 'DELETE' THEN

-- Вызов функции count\_doctors\_by\_specialization с передачей id\_specialization

PERFORM count\_doctors\_by\_specialization(NEW.id\_specialization);

END IF;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

CREATE TRIGGER count\_doctors\_trigger

AFTER INSERT OR UPDATE OR DELETE ON "Doctor"

FOR EACH ROW

EXECUTE FUNCTION count\_doctors\_trigger\_func();

-- Проверка

INSERT INTO "Doctor" ("Full name", "Gender", "Date of birth", "Table number", "Work phone number", "id\_specialization")

VALUES ('Платонова Ольга Сергеевна', 'ЖЕН', '1980-01-01', '12345', '12345', 1);

SELECT count\_doctors\_by\_specialization(1);

# Список литературы

* + - 1. Васильева М.А., Хобта Д.О., Фильтрация набора данных. Рекомендации по выполнению работы и перечень типовых заданий: Учебно-методическое пособие. Издание второе, исправленное и дополненное–М.:РУТ(МИИТ). 2023.–105с.
      2. Васильева М.А., Меркулов Д.А. Группировка и обобщение данных. Рекомендации по выполнению работы и перечень типовых заданий. Учебно-методическое пособие. М.:РУТ(МИИТ), 2023. 46–с.
      3. Васильева М.А., Ракинцев Н.А. Соединение данных из множества таблиц. Рекомендации по выполнению работы и перечень типовых заданий. Учебно-методическое пособие. М.:РУТ(МИИТ), 2023. 63–с.
      4. Балакина Е.П., Васильева М.А., Филипченко К.М. Информационное обеспечение систем управления. Методические указания к курсовому проектированию. Учебно-методическое пособие. Издание второе, исправленное и дополненное, 2023.102–с.