|  |
| --- |
| МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА» (РУТ (МИИТ))  ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Кафедра «Управление и защита информации» |
|  |
| **КУРСОВОЙ ПРОЕКТ**  **по дисциплине «Основы построения защищенных баз данных»** |
|  |
|  |
|  |
| Выполнил: ст. гр. ТКИ – 441  Чекан Ф.С. |
| Проверил: к.т.н., доц.  Васильева М.А. |
|  |
|  |
|  |
| Москва 2024 |

**Оглавление**

[Введение 3](#_Toc176184395)

[Проектирование базы данных 4](#_Toc176184396)

[Создание базы данных 8](#_Toc176184397)

[Создание таблиц 8](#_Toc176184398)

[Создание ограничений на пустые значения 9](#_Toc176184399)

[Добавление первичных ключей 10](#_Toc176184400)

[Добавление внешних ключей 10](#_Toc176184401)

[Содание ограничений на уникальное значение 11](#_Toc176184402)

[Заполнение базы данных 12](#_Toc176184403)

[Формирование запросов 13](#_Toc176184404)

[Формирование процедур 16](#_Toc176184405)

[Вывод 21](#_Toc176184406)

[Список литературы 22](#_Toc176184407)

Введение

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

В данном курсовом проекте ставится задача разработать РБД в СУБД PostgreSQL по теме «Больница», необходимо разработать скрипты на создание и заполнение РБД.

База данных создается для информационного обслуживания сотрудников больницы. В нашем случае – классическая городская клиническая больница.

Готовые запросы:

* Показывать пациентов в определённой палате.
* Показывать пациентов, которые закреплены за конкретным врачом.
* Показывать пациентов, которые находятся в определённом отделении.
* Показывать врачей по определённому отделению.
* Показывать врачей, которые на данный момент находятся на дежурстве.

Проектирование базы данных

Выделим базовые сущности базы данных «Больница»:

* Врачи;
* Пациенты;
* Палаты;
* Отделения;

Были созданы следующие сущности:

1. Таблица "Doctors" - содержит информацию о докторах.
2. Таблица "Patients" – содержит информацию о пациентах.
3. Таблица "Wards" - содержит информацию о палатах.
4. Таблица "Departments" - содержит информацию об отделениях.

На рисунке ниже (рисунок 1) представлена ER-диаграмма БД.

|  |
| --- |
|  |
| 1. – ER-диаграмма базы данных «Больница» |

1. - Схема отношения Врачи (Doctors)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор доктора | ID | int4 | Первичный ключ, обязательное поле |
| Идентификатор отделения | Department\_ID | Int4 | Внешний ключ, обязательное поле |
| Фамилия доктора | Last\_Name | varchar(200) | Обязательное поле |
| Имя доктора | First\_Name | varchar(200) | Обязательное поле |
| Отчество доктора | Patronymic | varchar(200) | Обязательное поле |
| Статус дежурства | Duty\_Status | boolean | Обязательное поле |

1. - Схема отношения Пациенты (Patients)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор пациента | ID | int4 | Первичный ключ, обязательное поле |
| Идентификатор палаты | Ward\_ID | Int4 | Внешний ключ, обязательное поле |
| Фамилия пациента | Last\_Name | varchar(200) | Обязательное поле |
| Имя пациента | First\_Name | varchar(200) | Обязательное поле |
| Отчество пациента | Patronymic | varchar(200) | Обязательное поле |

1. - Схема отношения Отделения (Departments)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор отделения | ­ ID | int4 | Первичный ключ, обязательное поле |
| Название отделения | Name | varchar(200) | Обязательное поле |

1. - Схема отношения Палаты (Wards)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор палаты (номер) | ID | int4 | Первичный ключ, обязательное поле |
| Идентификатор отделения | Department\_ID | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения Информационная таблица текущих больных (Info)

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор пациента | Patient\_ID | int4 | Внешний ключ, обязательное поле |
| Идентификатор врача | Doctor\_ID | int4 | Внешний ключ, обязательное поле |

Создание базы данных

Ниже представлены скрипты для создания базы данных и таблиц.

Создание базы данных «Больница»:

**create** **database** **"Hospital"**;

Создание таблиц

Создание таблицы Врачи:

**create table Doctors(**

**ID int,**

**Department\_ID int,**

**Last\_Name varchar(200),**

**First\_Name varchar (200),**

**Patronymic varchar(200),**

**Duty\_Status boolean**

**)**;

Создание таблицы Пациенты:

**create table Patients(**

**ID int,**

**Ward\_ID int,**

**Last\_Name varchar(200),**

**First\_Name varchar (200),**

**Patronymic varchar(200)**

**);**

Создание таблицы Отделения:

**create table Departments(**

**ID int,**

**Name varchar(200)**

**);**

Создание таблицы Палаты:

**create table Wards(**

**ID int,**

**Department\_ID int**

**);**

Создание таблицы с информацией о пациентах и врачах:

**create table Info(**

**Patient\_ID int,**

**Doctor\_ID int**

**);**

Создание ограничений на пустые значения

В каждой из таблиц добавим ограничения на пустые значения для столбцов, которые не могут являться пустыми.

**alter table doctors**

**alter column ID set not null,**

**alter column Department\_ID set not null,**

**alter column Last\_Name set not null,**

**alter column First\_Name set not null,**

**alter column Patronymic set not null,**

**alter column Duty\_Status set not null;**

**alter table patients**

**alter column ID set not null,**

**alter column Ward\_ID set not null,**

**alter column Last\_Name set not null,**

**alter column First\_Name set not null,**

**alter column Patronymic set not null;**

**alter table departments**

**alter column ID set not null,**

**alter column Name set not null;**

**alter table wards**

**alter column ID set not null,**

**alter column Department\_ID set not null;**

**alter table info**

**alter column Patient\_ID set not null,**

**alter column Doctor\_ID set not null;**

Добавление первичных ключей

Ниже представлены ограничение первичного ключа.

**alter table doctors**

**add constraint pk\_doctor primary key (ID);**

**alter table patients**

**add constraint pk\_patient primary key (ID);**

**alter table departments**

**add constraint pk\_department primary key (ID);**

**alter table wards**

**add constraint pk\_ward primary key (ID);**

Добавление внешних ключей

Ниже представлены связи между таблицами с помощью добавления внешних ключей.

**alter table doctors add constraint fk\_doctors\_departments foreign key (Department\_ID) references Departments(ID);**

**alter table patients add constraint fk\_patients\_wards foreign key (Ward\_ID) references Wards(ID);**

**alter table wards add constraint fk\_wards\_departments foreign key (Department\_ID) references Departments(ID);**

**alter table info add constraint fk\_info\_doctors foreign key (Doctor\_ID) references Doctors(ID), add constraint fk\_info\_patients foreign key (Patient\_ID) references Patients(ID);**

Содание ограничений на уникальное значение

Ниже представлены ограничения уникальности для полей.

**alter table doctors add constraint uk\_doctors\_id unique (ID);**

**alter table wards add constraint uk\_wards\_id unique (ID);**

**alter table departments add constraint uk\_departments\_id unique (ID);**

**alter table patients add constraint uk\_patients\_id unique (ID);**

Заполнение базы данных

Ниже представлены скрипты для заполнения базы данных «Больница».

**insert into departments(ID, Name) values (1, 'Хирургия'), (2, 'Травматология'), (3, 'Реанимация'), (4, 'Гнойная хирургия'), (5, 'Терапия'), (6, 'Гинекология');**

**insert into departments(ID, Name) values (1, 'Хирургия'), (2, 'Травматология'), (3, 'Реанимация'), (4, 'Гнойная хирургия'), (5, 'Терапия'), (6, 'Гинекология');**

**insert into doctors(ID, Department\_ID, Last\_Name, First\_Name, Patronymic, Duty\_Status) values (1, 1, 'Иванов', 'Пётр', 'Владимирович', false), (2, 6, 'Королёва', 'Наталья', 'Кирилловна', true), (3, 5, 'Лахмостов', 'Вячеслав', 'Викторович', true), (4, 3, 'Кудинов', 'Валерий', 'Михайлович', true), (5, 2, 'Криковцов', 'Евгений', 'Леонидович', true), (6, 1, 'Белоусов', 'Василий', 'Александрович', true);**

**insert into doctors(ID, Department\_ID, Last\_Name, First\_Name, Patronymic, Duty\_Status) values (7, 4, 'Усенко', 'Александр', 'Викторович', true);**

**insert into wards(ID, Department\_ID) values (1, 1), (2, 1), (3, 1), (4, 1), (5, 1), (6, 2), (7, 2), (8, 2), (9, 2), (10, 2), (11, 3), (12, 3), (13, 3), (14, 3), (15, 3), (16, 4), (17, 4), (18, 4), (19, 4), (20, 4), (21, 5), (22, 5), (23, 5), (24, 5), (25, 5), (26, 6), (27, 6), (28, 6), (29, 6), (30, 6);**

**insert into patients (ID, Ward\_ID, Last\_Name, First\_Name, Patronymic) values (1, 9, 'Корнелюк', 'Антон', 'Викторович'), (2, 25, 'Пудова', 'Виктория', 'Викторовна'), (3, 11, 'Захарова', 'Александра', 'Витальевна'), (4, 28, 'Хомякова', 'Алевтина', 'Евдокимовна'), (5, 16, 'Корнеев', 'Борис', 'Вячеславович'), (6, 30, 'Бондарева', 'Александра', 'Антоновна'), (7, 23, 'Ботвиньев', 'Никита', 'Олегович');**

**insert into info (Patient\_ID, Doctor\_ID) values (7, 3), (1, 5), (2, 3), (3, 4), (4, 2), (5, 7), (6, 2);**

Формирование запросов

1) Показывать пациентов в определённой палате.

--Показывать пациентов в определённой палате.

**CREATE OR REPLACE FUNCTION ShowPatientsInfo(in ward\_number int)**

**RETURNS TABLE (last\_name varchar (200), first\_name varchar (200), patronymic varchar (200)) AS $$**

**BEGIN**

**RETURN QUERY**

**SELECT Patients.last\_name, Patients.first\_name, Patients.patronymic**

**FROM Patients**

**WHERE ward\_number = Patients.ward\_id;**

**END;**

**$$ LANGUAGE plpgsql;**

|  |
| --- |
|  |
| 1. – Вывод пациентов в определённой палате |

2) Показывать пациентов, которые закреплены за конкретным врачом.

--Показывать пациентов, которые закреплены за конкретным врачом.

**CREATE OR REPLACE FUNCTION get\_patients\_by\_doctor(p\_doctor\_id INT)**

**RETURNS TABLE(patient\_id INT, last\_name VARCHAR, first\_name VARCHAR, patronymic VARCHAR) AS $$**

**BEGIN**

**RETURN QUERY**

**SELECT p.id, p.last\_name, p.first\_name, p.patronymic**

**FROM Patients p**

**INNER JOIN Info i ON p.id = i.patient\_id**

**WHERE i.doctor\_id = p\_doctor\_id;**

**END;**

**$$ LANGUAGE plpgsql;**

|  |
| --- |
|  |
| 1. – Вывод информации о пациентах, закреплённых за врачом |

3) Показывать пациентов, которые находятся в определённом отделении.

--Показывать пациентов, которые находятся в определённом отделении.

**CREATE OR REPLACE FUNCTION get\_patients\_by\_department(p\_department\_id INT)**

**RETURNS TABLE(patient\_id INT, last\_name VARCHAR, first\_name VARCHAR, patronymic VARCHAR) AS $$**

**BEGIN**

**RETURN QUERY**

**SELECT p.id, p.last\_name, p.first\_name, p.patronymic**

**FROM Patients p**

**INNER JOIN Wards w ON p.ward\_id = w.id**

**WHERE w.department\_id = p\_department\_id;**

**END;**

**$$ LANGUAGE plpgsql;**

|  |
| --- |
|  |
| 1. – Вывод пациентов в определённом отделении |

4) Показывать врачей по определённому отделению.

--Показать врачей по определённому отделению.

**CREATE OR REPLACE FUNCTION get\_doctors\_by\_department(p\_department\_id INT)**

**RETURNS TABLE(doctor\_id INT, last\_name VARCHAR, first\_name VARCHAR, patronymic VARCHAR) AS $$**

**BEGIN**

**RETURN QUERY**

**SELECT d.id, d.last\_name, d.first\_name, d.patronymic**

**FROM Doctors d**

**WHERE d.department\_id = p\_department\_id;**

**END;**

**$$ LANGUAGE plpgsql;**

|  |
| --- |
|  |
| 1. – Показать врачей по определённому отделению |

5) Показывать врачей, которые на данный момент находятся на дежурстве.

--Показать врачей, которые на данный момент находятся на дежурстве.

**CREATE OR REPLACE FUNCTION get\_doctors\_on\_duty()**

**RETURNS TABLE(doctor\_id INT, last\_name VARCHAR, first\_name VARCHAR, patronymic VARCHAR) AS $$**

**BEGIN**

**RETURN QUERY**

**SELECT d.id, d.last\_name, d.first\_name, d.patronymic**

**FROM Doctors d**

**WHERE d.duty\_status = TRUE;**

**END;**

**$$ LANGUAGE plpgsql;**

|  |
| --- |
|  |
| 1. – Показать врачей на дежурстве |

Формирование процедур

1. Процедура добавления данных в таблицу Doctors:

-- Создание процедуры для вставки данных в таблицу Doctors

**CREATE OR REPLACE PROCEDURE add\_doctor(**

**p\_id INT,**

**p\_last\_name VARCHAR,**

**p\_first\_name VARCHAR,**

**p\_patronymic VARCHAR,**

**p\_department\_id INT,**

**p\_duty\_status BOOLEAN**

**)**

**LANGUAGE plpgsql**

**AS $$**

**BEGIN**

**INSERT INTO Doctors (id, last\_name, first\_name, patronymic, department\_id, duty\_status)**

**VALUES (p\_id, p\_last\_name, p\_first\_name, p\_patronymic, p\_department\_id, p\_duty\_status);**

**END;**

**$$;**

**CALL add\_doctor(45, 'Иванов', 'Иван', 'Иванович', 3, TRUE);**

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Doctors |

1. Процедура добавления данных в таблицу Patients:

-- Создание процедуры для вставки данных в таблицу Patients

**CREATE OR REPLACE PROCEDURE add\_patient(**

**p\_id INT,**

**p\_last\_name VARCHAR,**

**p\_first\_name VARCHAR,**

**p\_patronymic VARCHAR,**

**p\_ward\_id INT**

**)**

**LANGUAGE plpgsql**

**AS $$**

**BEGIN**

**INSERT INTO Patients (id, last\_name, first\_name, patronymic, ward\_id)**

**VALUES (p\_id, p\_last\_name, p\_first\_name, p\_patronymic, p\_ward\_id);**

**END;**

**$$;**

**CALL add\_patient(89, 'Петров', 'Артемий', 'Петрович', 7);**

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Patients |

1. Процедура добавления данных в таблицу Departments:

**CREATE OR REPLACE PROCEDURE add\_department(**

**p\_id INT,**

**p\_name VARCHAR**

**)**

**LANGUAGE plpgsql**

**AS $$**

**BEGIN**

**INSERT INTO Departments (id, name)**

**VALUES (p\_id, p\_name);**

**END;**

**$$;**

**CALL add\_department(36, 'Иммунология');**

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Departments |

1. Процедура добавления данных в таблицу Wards:

**CREATE OR REPLACE PROCEDURE add\_ward(**

**p\_id INT,**

**p\_department\_id INT**

**)**

**LANGUAGE plpgsql**

**AS $$**

**BEGIN**

**INSERT INTO Wards (id, department\_id)**

**VALUES (p\_id, p\_department\_id);**

**END;**

**$$;**

**CALL add\_ward(99, 5);**

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Wards |

1. Процедура добавления данных в таблицу Info:

**CREATE OR REPLACE PROCEDURE add\_info(**

**p\_patient\_id INT,**

**p\_doctor\_id INT**

**)**

**LANGUAGE plpgsql**

**AS $$**

**BEGIN**

**INSERT INTO Info (patient\_id, doctor\_id)**

**VALUES (p\_patient\_id, p\_doctor\_id);**

**END;**

**$$;**

**CALL add\_info(4, 7);**

|  |
| --- |
|  |
| 1. – Добавления данных в таблицу Info |

Вывод

В результате создания курсового проекта была спроектирована, создана и заполнена база данных для информационного обслуживания больницы.

Были изучены операторы SQL на примере СУБД PostgreSQL, необходимые для фильтрации набора данных; операторы, необходимые для работы с данными, находящимися в разных таблицах, получен навык создания запросов на соединение данных из множества таблиц.

Список литературы

1. 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.
2. Васильева М. А., Степанов Д. Е. Операторы языка определения данных. Учебно– методическое пособие. – М.: РУТ (МИИТ). 2024. – 33с
3. Васильева М. А., Балакина Е. П., Васильева М. В. Соединение данных из множества таблиц. Рекомендации по выполнению работы и перечень типовых заданий. Учебно–методическое пособие. Издание второе, исправленное. – М.: РУТ (МИИТ).2024. – 72с.
4. Васильева М. А, Филипченко К. М., Пугачёв В. А. Фильтрация набора данных. Рекомендации по выполнению работы и перечень типовых заданий: Учебно-методическое пособие. Издание третье, исправленное и дополненное. Москва–2024, 133 с.
5. Васильева М. А., Кузнецов К. К. Операторы языка управления данными. Учебно– методическое пособие. – М.: РУТ (МИИТ).2024. – 36с.