**Введение**

Разработка информационной системы требует создания в памяти компьютера модели внешнего мира, обновляемой в режиме реального времени, с использованием единого хранилища - базы данных. База данных (БД) представляет собой совокупность специальным образом организованных данных, хранимых в памяти вычислительной системы и отображающих состояние объектов и их взаимосвязей в рассматриваемой предметной области. Предметная область - часть реального мира, подлежащая изучению с целью организации управления и, в конечном счете, автоматизации.

Для работы с базой данных необходима СУБД (система управления базами данных), т.е. программа, которая берет на себя все заботы, связанные с доступом к данным. Система управления базами данных - это комплекс языковых и программных средств, предназначенный для создания, ведения и совместного использования БД многими пользователями. Обычно СУБД различают по используемой модели данных. Так, СУБД, основанные на использовании реляционной модели данных, называют реляционными СУБД. Реляционная модель данных – это логическая модель данных основанная на математической теории отношений. Отношение (relation) представляет собой множество элементов, называемых кортежами. Наглядной формой представления отношения является привычная для человеческого восприятия двумерная таблица. Строкам таблицы соответствуют кортежи, а столбцам - атрибуты отношения.

В данной работе в качестве предметной области рассматривается стационар больницы. Реляционная база данных больницы хранит полную информацию о стационарных больных, а также структуре больницы. В качестве основных объектов предметной области выбраны следующие сущности: физическое лицо, пол человека, врач, специализация врача, отделение больницы, лечащий врач пациента, пациент, родственник пациента, медицинская карта, диагноз, обследование, назначение, лечение. Создание, заполнение таблиц реляционной базы данных, а также осуществление обращений к БД выполнено при помощи СУБД PostgreSQL.

**Ход работы**

CREATE TABLE genderskii (

idgender integer,

gender varchar(10) UNIQUE,

PRIMARY KEY (idgender)

);

CREATE TABLE personskii (

idperson integer,

surname varchar(30),

pname varchar(30),

otch varchar(30),

idgender integer,

city varchar(30),

street varchar(30),

nhouse varchar(5),

kv varchar(5),

korp varchar(2),

dob varchar(30),

phone varchar(20),

passport text,

PRIMARY KEY (idperson),

FOREIGN KEY (idgender)

REFERENCES genderskii (idgender)

);

CREATE TABLE patientskii (

idpatient integer,

idperson integer,

polic varchar(20),

job varchar(50),

weight numeric CHECK (weight>0),

height numeric CHECK (height>0),

PRIMARY KEY (idpatient),

FOREIGN KEY (idperson)

REFERENCES personskii (idperson)

);

CREATE TABLE kinshipkii (

idkinship integer,

idpatient integer,

idperson integer,

PRIMARY KEY (idkinship),

FOREIGN KEY (idpatient)

REFERENCES patientskii (idpatient)

ON DELETE CASCADE,

FOREIGN KEY (idperson)

REFERENCES personskii (idperson)

ON DELETE SET NULL

);

CREATE TABLE departmentskii (

iddepartment integer,

department varchar(50) UNIQUE,

addressofdep text,

PRIMARY KEY (iddepartment)

);

CREATE TABLE specialtieskii (

idspecialty integer,

specialty varchar(50),

PRIMARY KEY (idspecialty)

);

CREATE TABLE doctorskii (

iddoctor integer,

idperson integer,

iddepartment integer,

idspecialty integer,

PRIMARY KEY (iddoctor),

FOREIGN KEY (idperson)

REFERENCES personskii (idperson)

ON DELETE CASCADE,

FOREIGN KEY (iddepartment)

REFERENCES departmentskii (iddepartment)

ON DELETE SET NULL,

FOREIGN KEY (idspecialty)

REFERENCES specialtieskii (idspecialty)

);

CREATE TABLE diagnoseskii (

iddiagnosis integer,

diagnosis text,

PRIMARY KEY (iddiagnosis)

);

CREATE TABLE medcardskii (

idmedcard integer,

idpatient integer,

iddiagnosis integer,

result varchar(50),

datein varchar(50) NOT NULL,

dateout varchar(50),

PRIMARY KEY (idmedcard),

FOREIGN KEY (idpatient)

REFERENCES patientskii (idpatient)

ON DELETE CASCADE,

FOREIGN KEY (iddiagnosis)

REFERENCES diagnoseskii (iddiagnosis)

);

CREATE TABLE treatmentskii (

idtreatment integer,

idmedcard integer,

iddoctor integer,

PRIMARY KEY (idtreatment),

FOREIGN KEY (idmedcard)

REFERENCES medcardskii (idmedcard)

ON DELETE CASCADE,

FOREIGN KEY (iddoctor)

REFERENCES doctorskii (iddoctor)

ON DELETE SET NULL

);

CREATE TABLE surveyskii (

idsurvey integer,

idmedcard integer,

iddoctor integer,

survey varchar(50),

dateofsurvey varchar(50) NOT NULL,

cabinet varchar(10),

PRIMARY KEY (idsurvey),

FOREIGN KEY (idmedcard)

REFERENCES medcardskii (idmedcard)

ON DELETE CASCADE,

FOREIGN KEY (iddoctor)

REFERENCES doctorskii (iddoctor)

);

CREATE TABLE prescriptionskii (

idprescription integer,

prescription text,

dose varchar(50),

PRIMARY KEY (idprescription)

);

CREATE TABLE therapieskii (

idtherapy integer,

idsurvey integer,

idprescription integer,

PRIMARY KEY (idtherapy),

FOREIGN KEY (idsurvey)

REFERENCES surveyskii (idsurvey)

ON DELETE CASCADE,

FOREIGN KEY (idprescription)

REFERENCES prescriptionskii (idprescription)

);

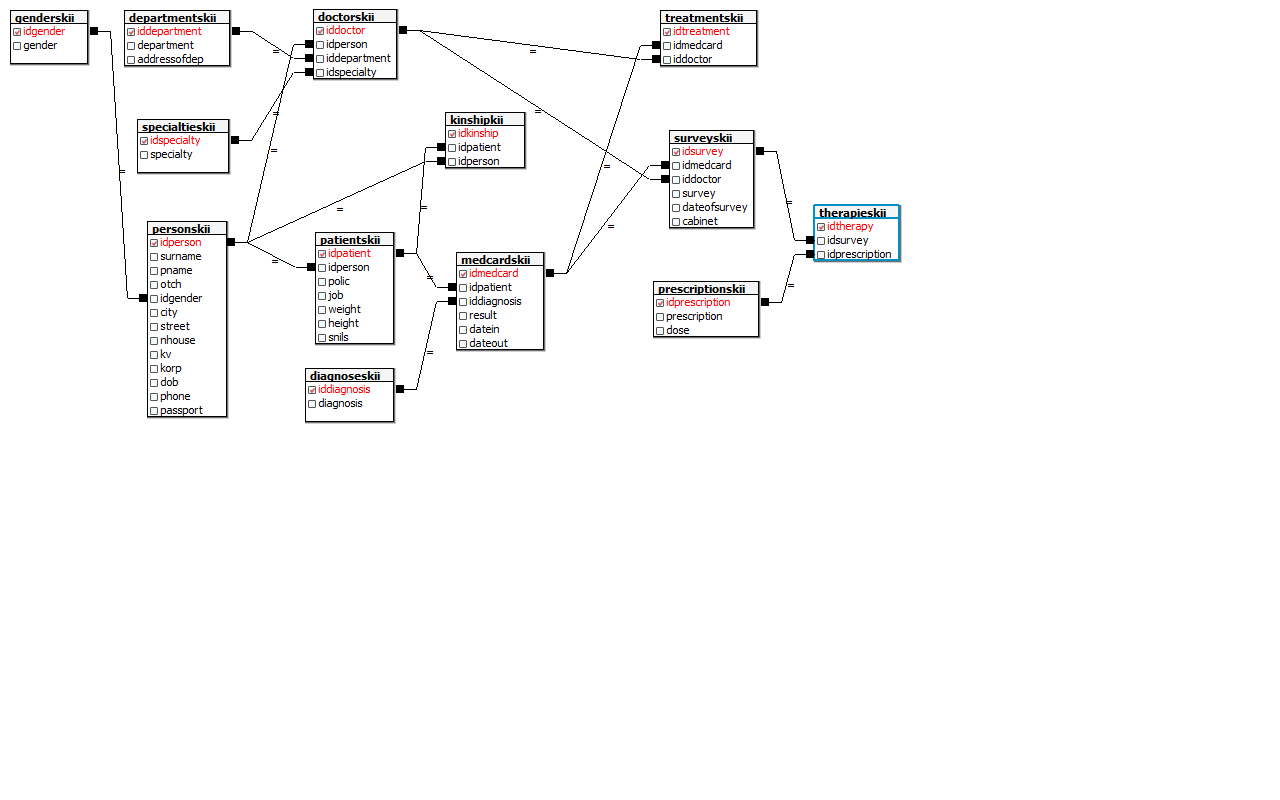


Рисунок 1 - Связь в БД

1) Модификация таблицы добавлением какого-либо атрибута.

*Добавить в таблицу, содержащую информацию о пациентах, столбец для данных СНИЛС и заполнить его.*

ALTER TABLE patientskii ADD COLUMN snils varchar(15);

UPDATE patientskii SET snils='160-165-158 42' WHERE polic='1987456428';

UPDATE patientskii SET snils='160-125-169 43' WHERE polic='5637829381';

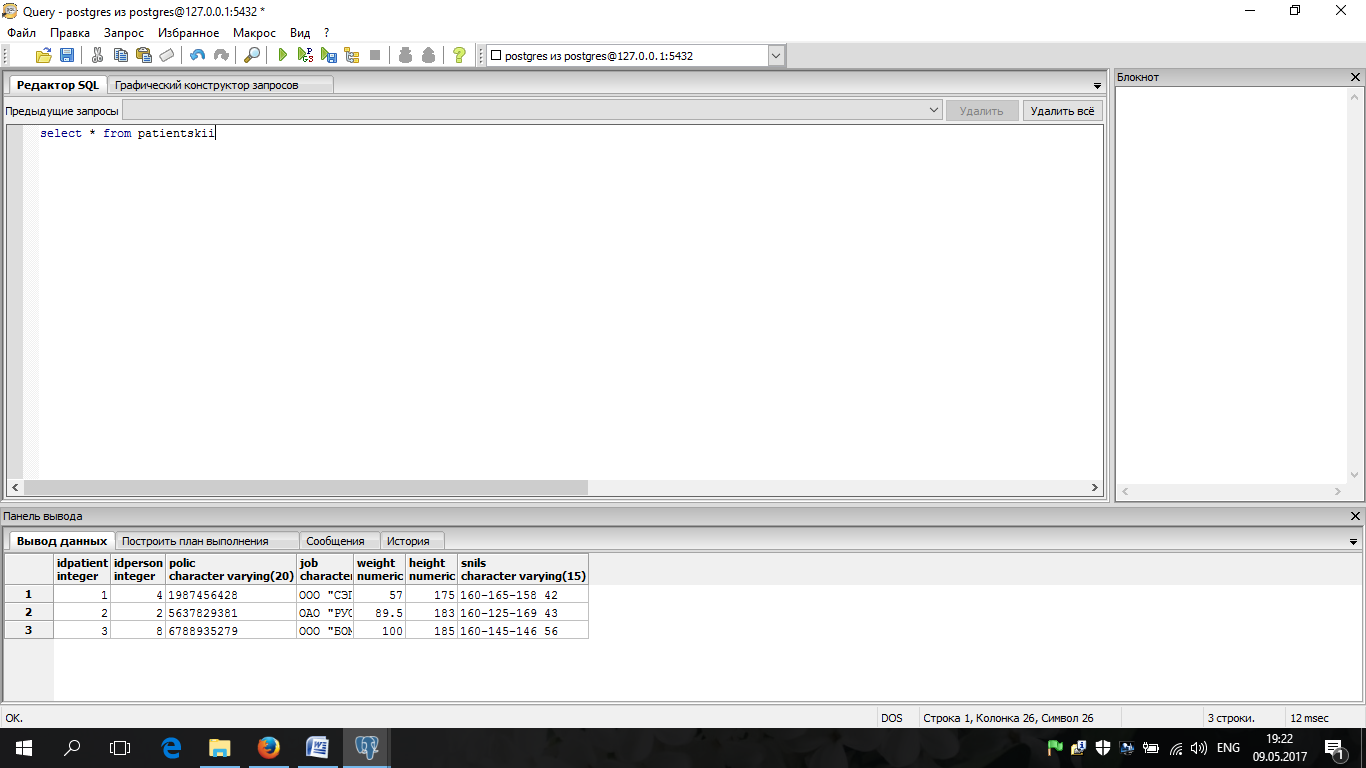
UPDATE patientskii SET snils='160-145-146 56' WHERE polic='6788935279';

2) Выборка всех данных из таблицы.

*Проверим информацию, содержащуюся в таблице пациентов. Выведем все кортежи patientskii.*

SELECT \* FROM patientskii

Вывод данных:

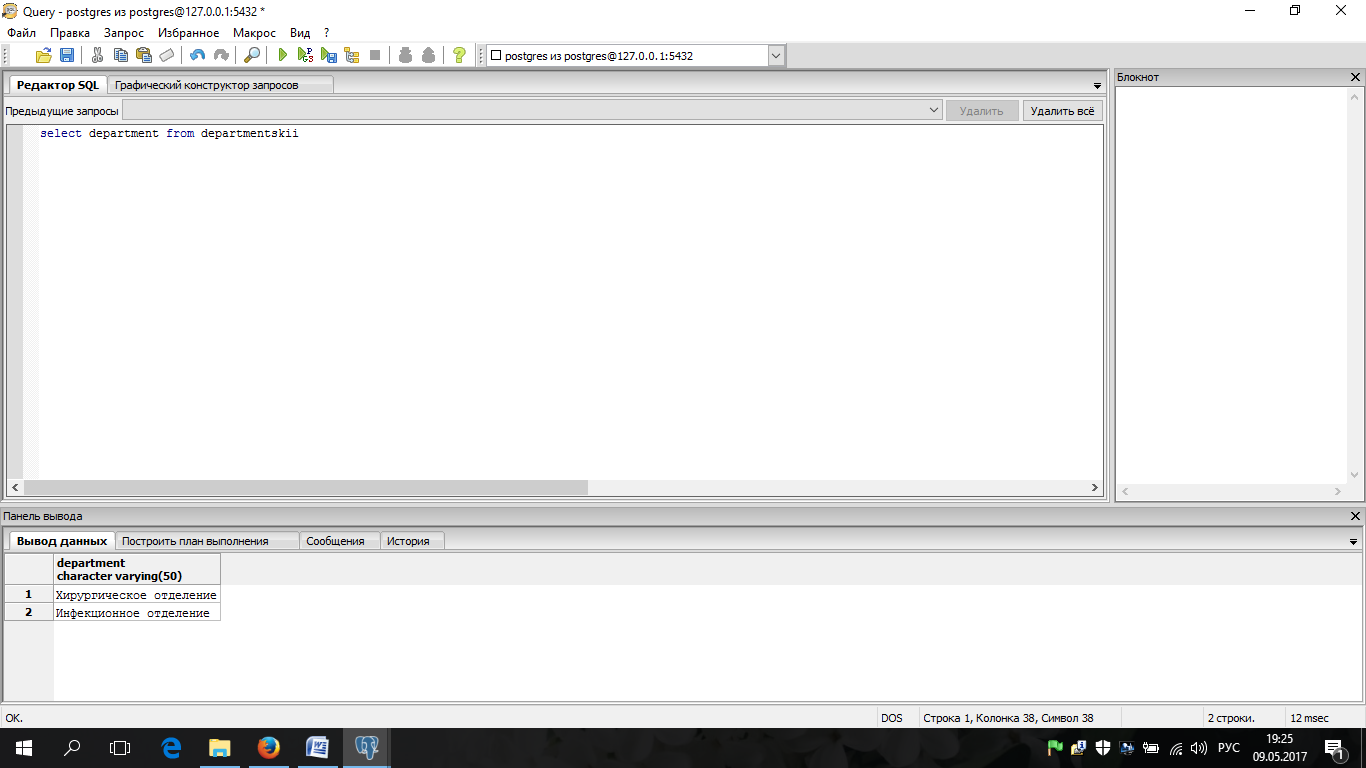


3)Выборка различных значений какого-либо столбца таблицы.

*Вывести список всех отделений больницы, внесенных в БД.*

SELECT department FROM departmentskii

Вывод данных:



4) Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному диапазону.

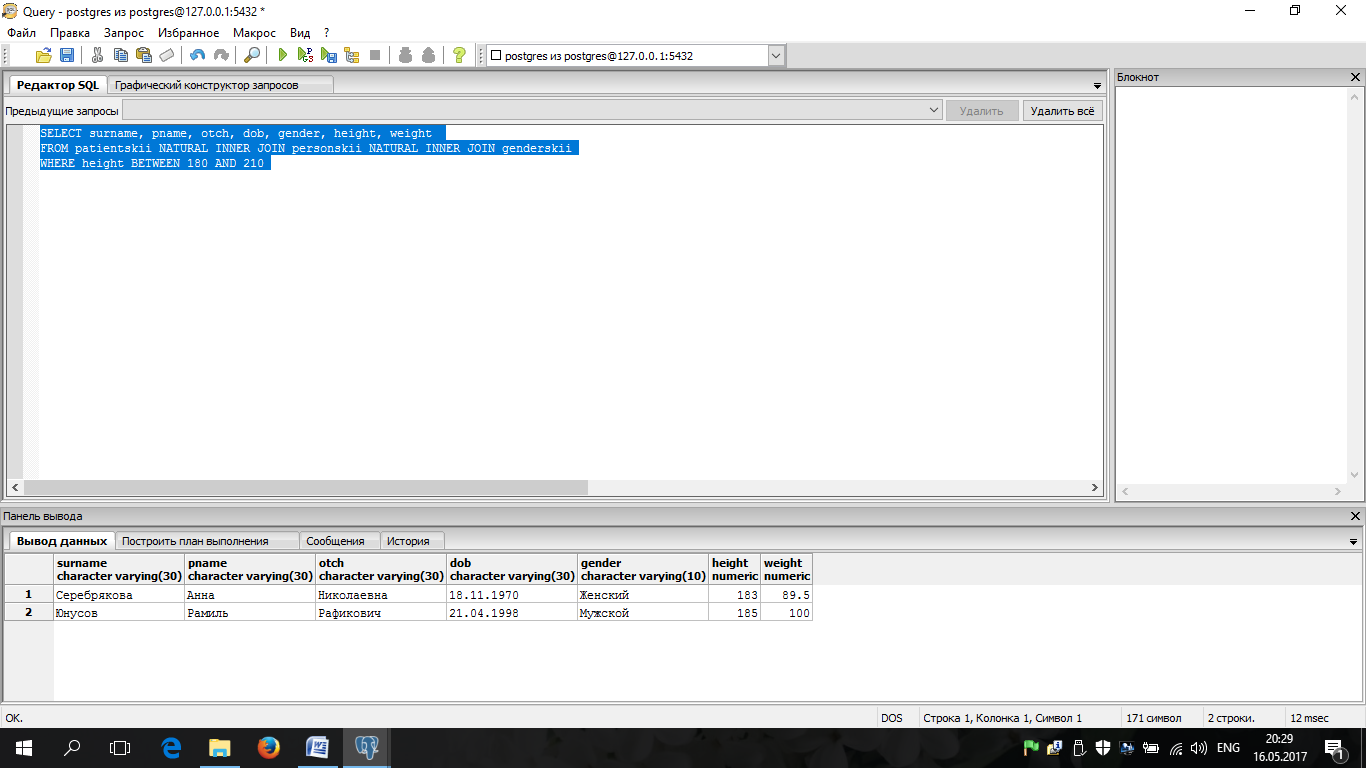
*Вывести список пациентов с ростом от 180 до 210 см.*

SELECT surname, pname, otch, dob, gender, height, weight

FROM patientskii NATURAL INNER JOIN personskii NATURAL INNER JOIN genderskii

WHERE height BETWEEN 180 AND 210

Вывод данных:



5) Выборка строк таблицы, где значения какого-либо атрибута принадлежат заданному набору значений.

*Вывести список пациентов, которым были сделаны заданные в запросе назначения.*

SELECT surname, pname, otch, prescription, dose

FROM prescriptionskii

NATURAL INNER JOIN therapieskii

NATURAL INNER JOIN surveyskii

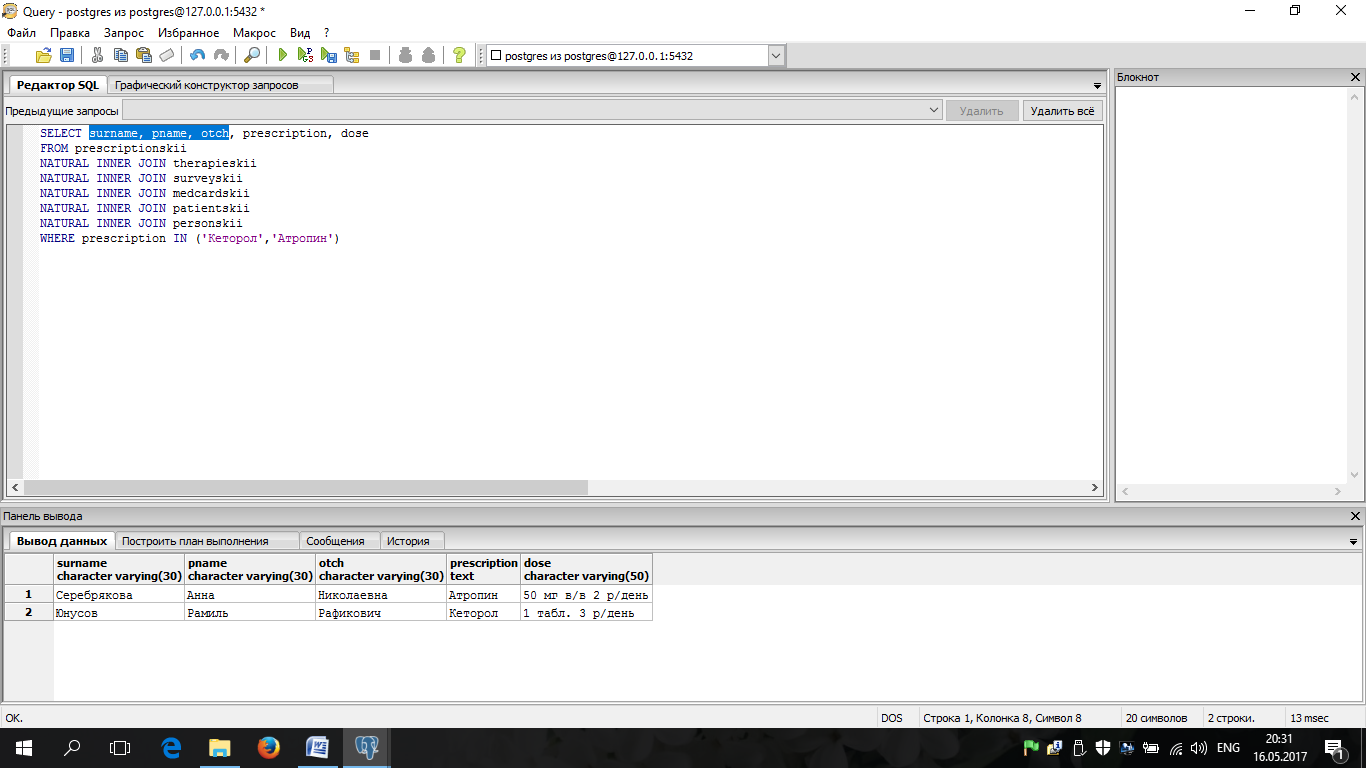
NATURAL INNER JOIN medcardskii

NATURAL INNER JOIN patientskii

NATURAL INNER JOIN personskii

WHERE prescription IN ('Кеторол','Атропин')

Вывод данных:



6) Выборка строк таблицы, где значения какого-либо атрибута соответствуют заданному шаблону.

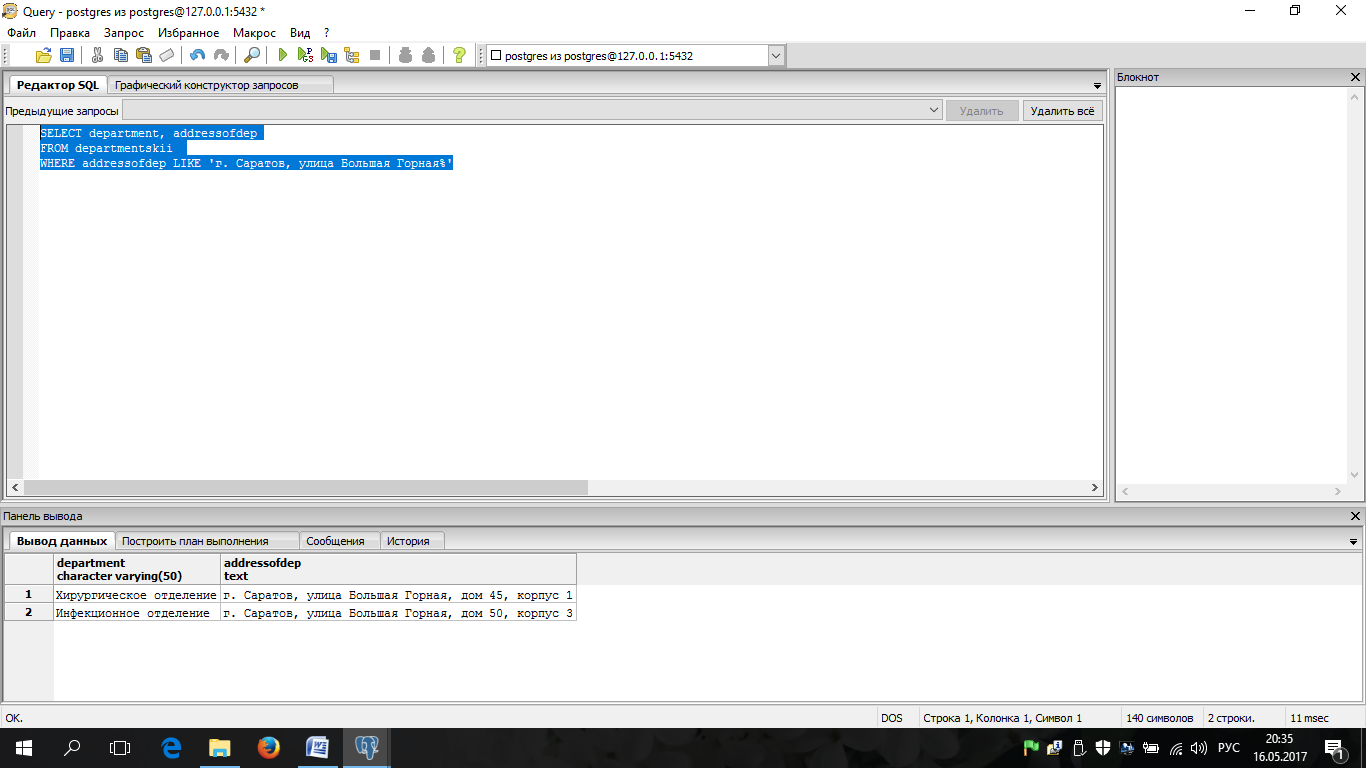
*Вывести список отделений, находящихся в определенном районе города.*

SELECT department, addressofdep

FROM departmentskii

WHERE addressofdep LIKE 'г. Саратов, улица Большая Горная%'

Вывод данных:



7) Выборка строк таблицы, где значения какого-либо атрибута не пусто.

*Подсчитать количество больных, находящихся под наблюдением (не выписанных на данный момент).*

SELECT COUNT(idmedcard) FROM medcardskii WHERE dateout IS NOT NULL

8) Сортировка строк таблицы по двум ключам сортировки.

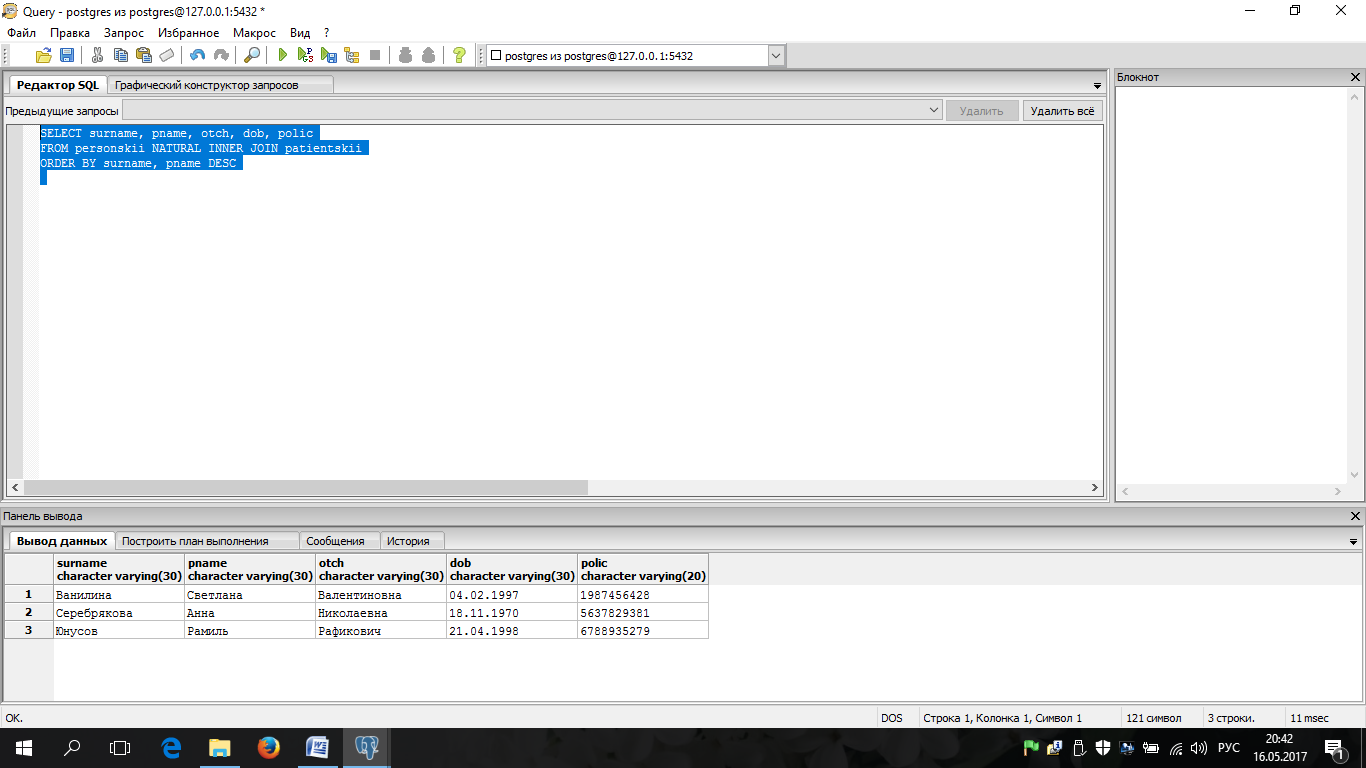
*Вывести данные о людях, проходивших лечение, отсортировав их по фамилиии и имени.*

SELECT surname, pname, otch, dob, polic

FROM personskii NATURAL INNER JOIN patientskii

ORDER BY surname, pname DESC

Вывод данных:



9) Внутреннее естественное соединение таблиц.

*Получить информацию о количестве стационарных наблюдений в больнице пациента с указанным номером медицинского полиса.*

SELECT COUNT(idmedcard) AS numofstays

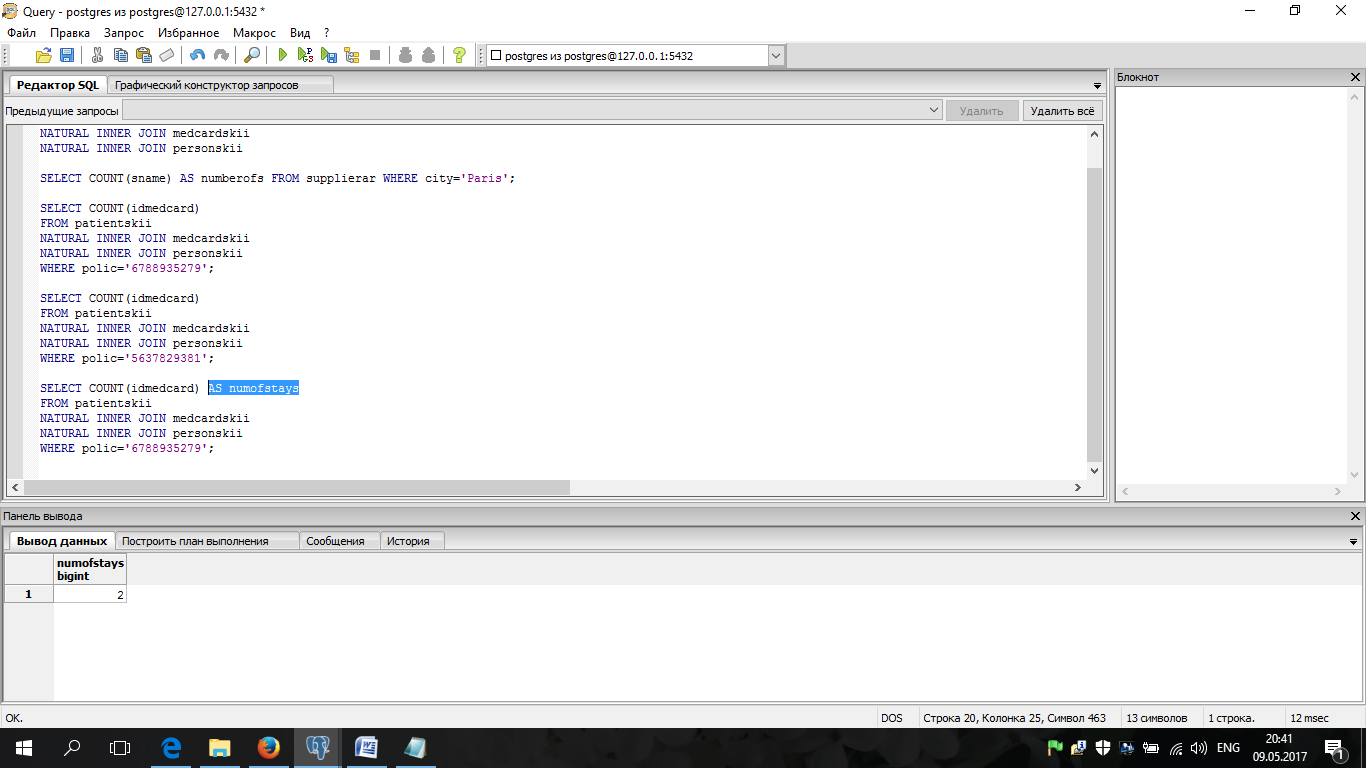
FROM patientskii

NATURAL INNER JOIN medcardskii

NATURAL INNER JOIN personskii

WHERE polic='6788935279';

Вывод данных:



10) Правое соединение таблиц.

SELECT \* FROM personskii RIGHT OUTER JOIN patientskii using (idperson)

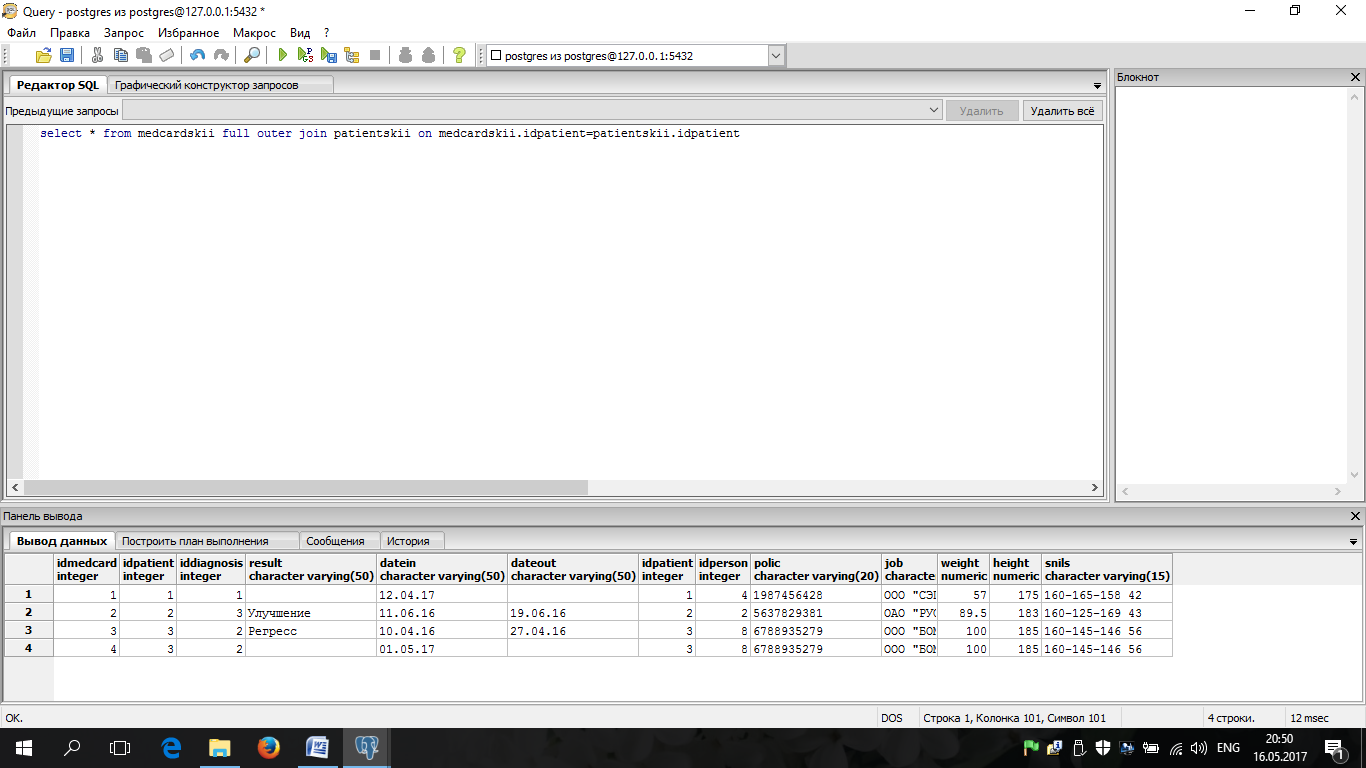
11) Левое соединение таблиц.

SELECT \* FROM personskii NATURAL LEFT OUTER JOIN patientskii

12) Объединение двух таблиц.

SELECT \* FROM medcardskii FULL OUTER JOIN patientskii ON medcardskii.idpatient=patientskii.idpatient

Вывод данных:



13) Группировка записей по двум или более полям.

*Узнать, сколько врачей определенной специальности работают в каждом отделении.*

SELECT department, specialty, COUNT(iddoctor)

FROM doctorskii

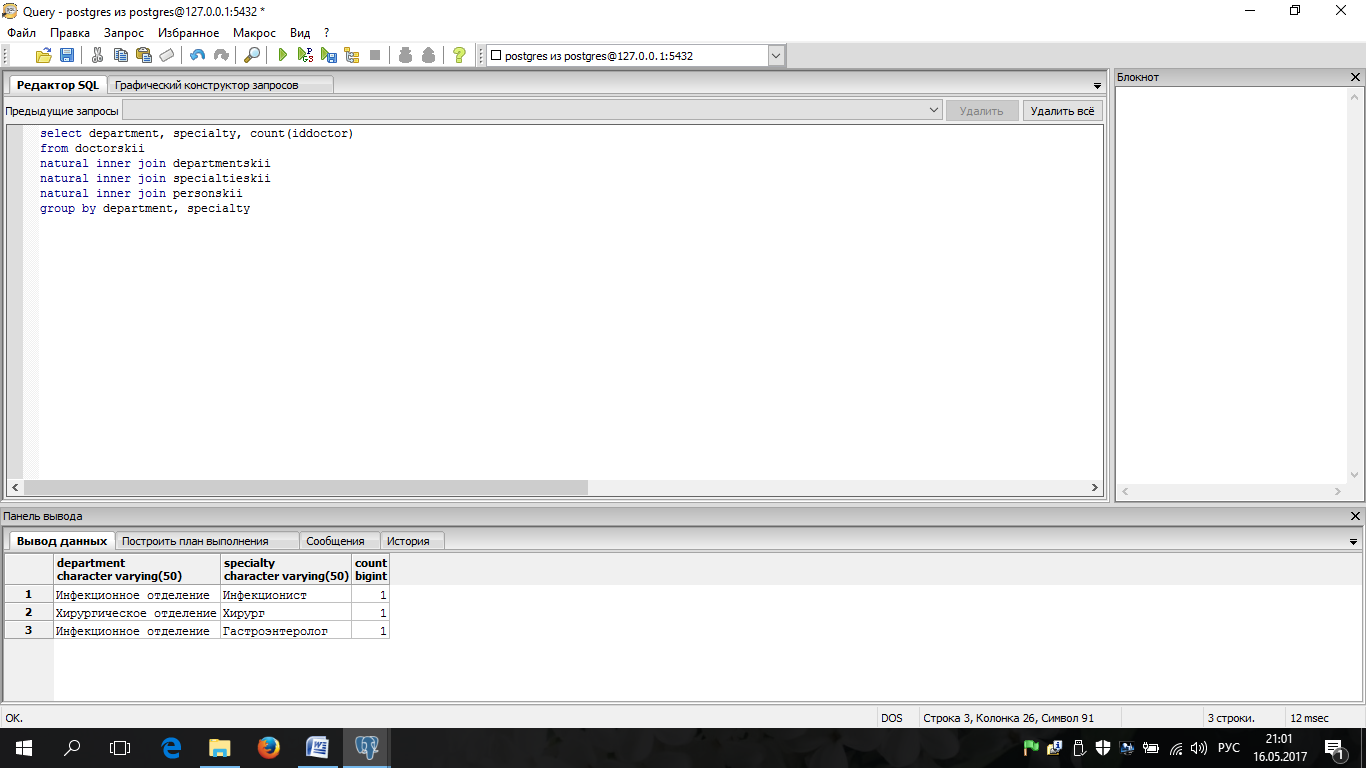
NATURAL INNER JOIN departmentskii

NATURAL INNER JOIN specialtieskii

NATURAL INNER JOIN personskii

GROUP BY department, specialty

Вывод данных:

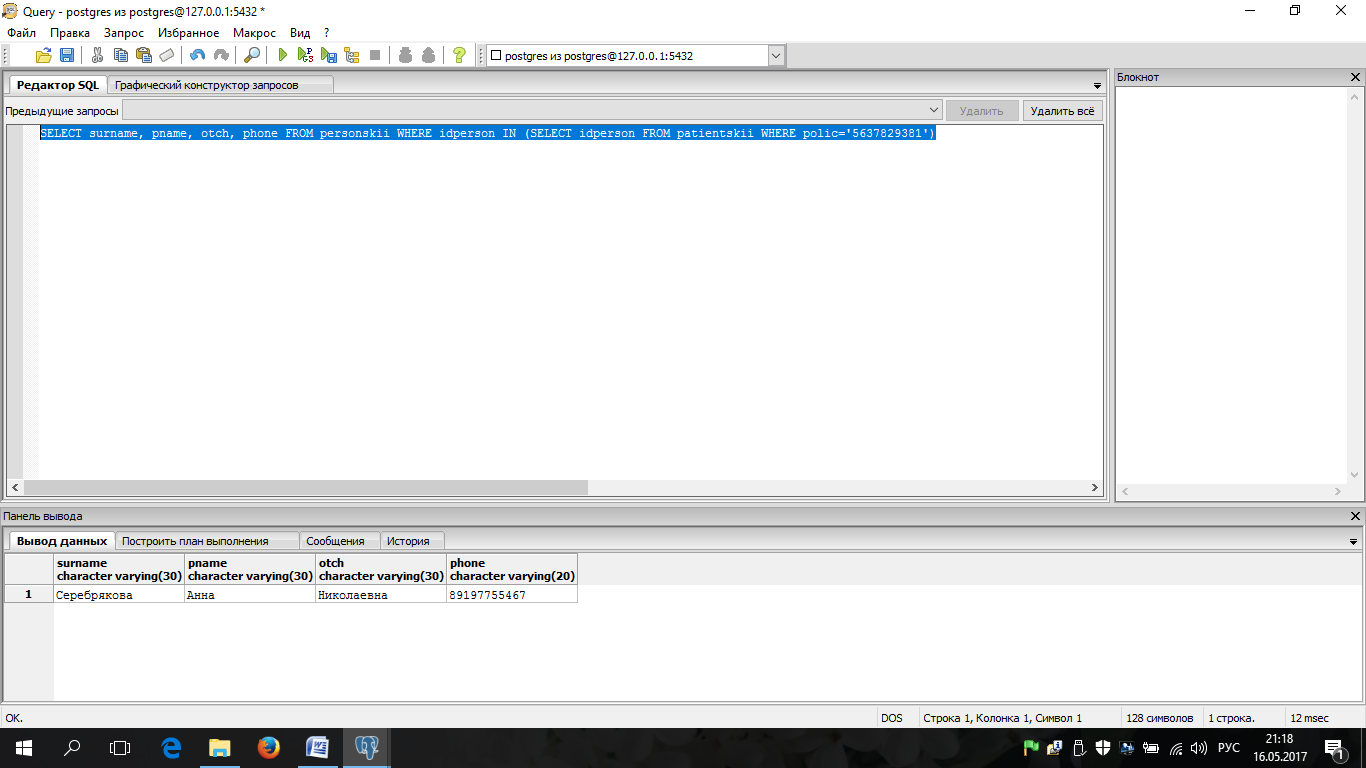


14) Вложенный подзапрос.

*Отобразить контактные данные пациента с данным номером полиса.*

SELECT surname, pname, otch, phone FROM personskii WHERE idperson IN (SELECT idperson FROM patientskii WHERE polic='5637829381')

Вывод данных:



15) Создание представлений.

*Создать представление, содержащее информацию о схеме лечения текущих пациентов.*

CREATE VIEW therapy AS

SELECT surname, pname, otch, prescription, dose, survey as basis, datein

FROM therapieskii

NATURAL INNER JOIN prescriptionskii

NATURAL INNER JOIN surveyskii

NATURAL INNER JOIN medcardskii

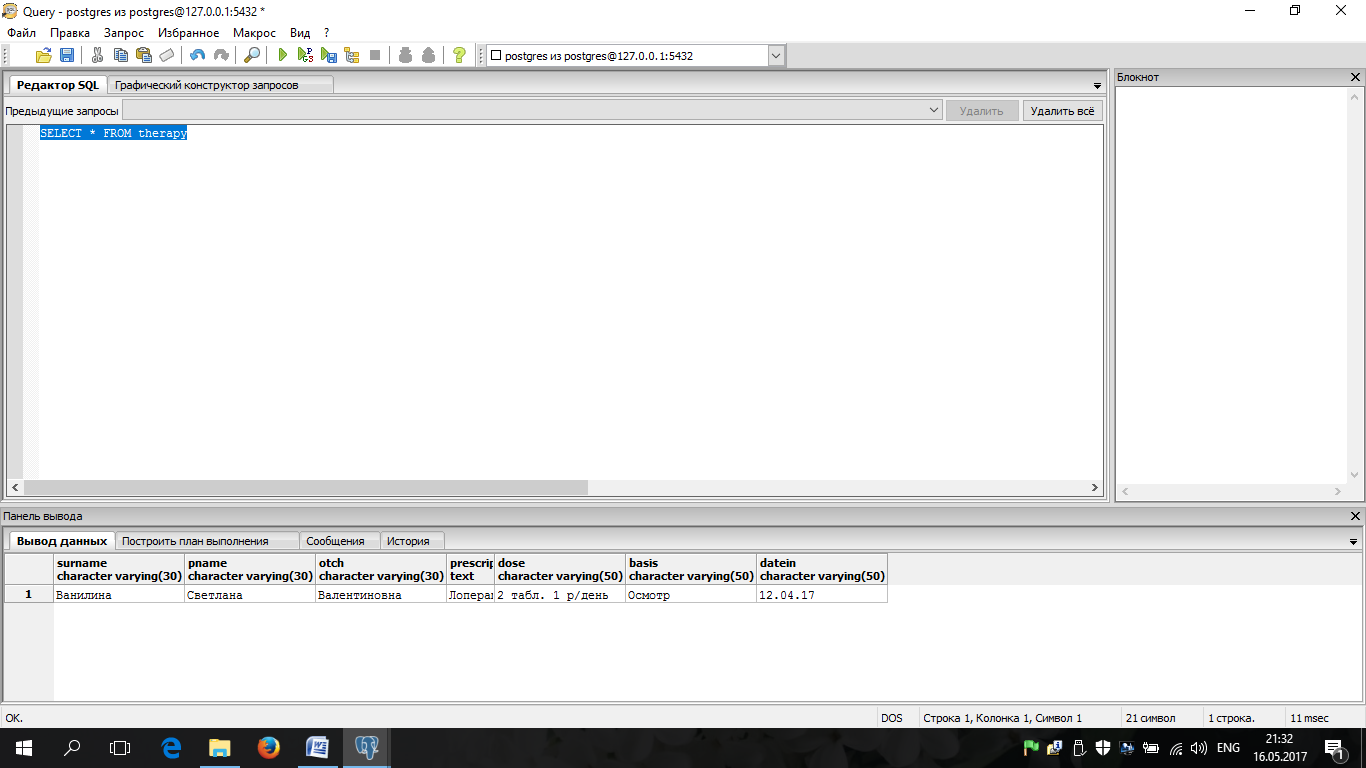
NATURAL INNER JOIN patientskii

NATURAL INNER JOIN personskii

WHERE dateout IS NULL

SELECT \* FROM therapy

Вывод данных:



**Заключение**

В ходе работы была создана модель базы данных для информационной системы стационара больницы. Предметная область была представлена множеством фрагментов. Каждый фрагмент предметной области характеризуется множеством объектов и процессов, использующих объекты. С помощью одной таблицы реляционной базы данных удобно описывать простейший вид связей между данными, а именно: деление одного объекта, информация о котором хранится в таблице, на множество подобъектов, каждому из которых соответствует строка или запись таблицы. Однако стоит заметить, что в рамках одной таблицы не всегда удается описать более сложные логические структуры данных из предметной области. Из этого следует вывод об удобстве применения связывания таблиц в рамках реляционной базы данных.

INSERT INTO genderskii VALUES

(1, 'Женский'),

(2, 'Мужской')

INSERT INTO personskii VALUES

(1, 'Иванов', 'Иван', 'Иванович', 2, 'г.Саратов', 'Электронная', '11', '67', NULL, '15.08.1996', '89097655467', '6310 554898'),

(2, 'Серебрякова', 'Анна', 'Николаевна', 1, 'г.Маркс', 'Мамонтова', '14а', '34', NULL, '18.11.1970', '89197755467', '6313 778894'),

(3, 'Пиденко', 'Варвара', 'Владимировна', 1, 'г.Саратов', 'Степная', '28', '11', NULL, '06.06.1998', '89177653367', '6311 745839'),

(4, 'Ванилина', 'Светлана', 'Валентиновна', 1, 'г.Саратов', 'Тархова', '4', '44', NULL, '04.02.1997', '89270585547', '6310 595732'),

(5, 'Попов', 'Артём', 'Олегович', 2, 'г.Аткарск', 'Сапожникова', '3', '66', NULL, '20.08.1996', '89157845567', '6313 559585'),

(6, 'Милонова', 'Арина', 'Викторовна', 1, 'г.Энгельс', '2-я Электронная', '13', '34', NULL, '19.09.1970', '89297533469', '6310 304896'),

(7, 'Пантелеева', 'Марина', 'Васильевна', 1, 'г.Саратов', 'Рахова', '51', '112', NULL, '28.01.1996', '89273834675', '6310 205673'),

(8, 'Юнусов', 'Рамиль', 'Рафикович', 2, 'г.Саратов', 'Зыбина', '5', '87', NULL, '21.04.1998', '89077623467', '6311 783267'),

(9, 'Петров', 'Семён', 'Витальевич', 2, 'г.Маркс', 'Земная', '1', '6', '2', '13.09.1980', '89197425347', '6313 578858')

INSERT INTO patientskii VALUES

(1, 4, '1987456428', 'ООО "СЭПО"', '57', '175'),

(2, 2, '5637829381', 'ОАО "РУССКИЕ МЕДВЕДИ"', '89.5', '183'),

(3, 8, '6788935279', 'ООО "БОМЖ"', '100', '185')

INSERT INTO kinshipkii VALUES

(1, 1, 5),

(2, 2, 9),

(3, 3, 1)

INSERT INTO departmentskii VALUES

(1, 'Хирургическое отделение', 'г. Саратов, улица Большая Горная, дом 45, корпус 1'),

(2, 'Инфекционное отделение', 'г. Саратов, улица Большая Горная, дом 50, корпус 3')

INSERT INTO specialtieskii VALUES

(1, 'Хирург'),

(2, 'Гастроэнтеролог'),

(3, 'Терапевт'),

(4, 'Инфекционист')

INSERT INTO doctorskii VALUES

(1, 3, 1, 1),

(2, 6, 2, 4),

(3, 7, 2, 2)

INSERT INTO diagnoseskii VALUES

(1, 'Сальмонеллез'),

(2, 'Киста гипофиза'),

(3, 'Отек Квинке')

INSERT INTO medcardskii VALUES

(1, 1, 1, NULL, '12.04.17', NULL),

(2, 2, 3, 'Улучшение', '11.06.16', '19.06.16'),

(3, 3, 2, 'Регресс', '10.04.16', '27.04.16'),

(4, 3, 2, NULL, '01.05.17', NULL)

INSERT INTO treatmentskii VALUES

(1, 1, 2),

(2, 2, 2),

(3, 3, 1)

INSERT INTO surveyskii VALUES

(1, 1, 2, 'Осмотр', '13.04.17', 8),

(2, 1, 3, 'ФГС', '14.04.17', 10),

(3, 2, 2, 'Осмотр', '11.06.16', 14),

(4, 2, 2, 'УЗИ','12.06.16', 1),

(5, 3, 1, 'МРТ', '15.04.16', 15)

INSERT INTO prescriptionskii VALUES

(1, 'Физиотерапия', '30 мин. 1 р/день'),

(2, 'Атропин', '50 мг в/в 2 р/день'),

(3, 'Кеторол', '1 табл. 3 р/день'),

(4, 'Лоперамид', '2 табл. 1 р/день')

INSERT INTO therapieskii VALUES

(1, 5, 1),

(2, 1, 4),

(3, 3, 2),

(4, 5, 3)