Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Кафедра экономической информатики

Лабораторная работа №2

«Проектирование базы данных»

Вариант №7

Выполнил: студент гр.

914302 Горлачёва А.О.

Проверил: Лукашевич А.Э.

Минск 2022

1. **Цель работы**

Изучение структурированного языка запросов Transact - SQL, являющегося основой системы программирования SQL Server, и приобретение навыков применение инструментальных средств разработки и программирования объектов создаваемых баз данных. Изучить SQL-операторы для работы с таблицами и индексами. Изучить sql- команды для создания, изменения и удаления таблиц. Изучить используемые в SQL Server типы ограничений. Изучить SQL-операторы для работы с ограничениями.

1. **Задача**

1. Создать **на языке Transact-SQL** файл базы данных согласно номеру варианта. База данных разрабатывается на основе спроектированной концептуальной модели данных в лаб.№1.

**Создать программно на языке SQL** все таблицы, с указанием первичных и внешних ключей и ограничения целостности.

Заполнить таблицы данными по 5 записей в каждой.

2. Для созданной базы данных, согласно номеру варианта, самостоятельно создать **на языке Transact-SQL 15 запросов с отбором строк по условию:**

* 3 простейших запроса с использованием операторов сравнения;
* 3 запроса с использованием логических операторов AND, OR и NOT;
* 2 запроса на использование комбинации логических операторов;
* 2 запроса на использование выражений над столбцами;
* 2 запроса с проверкой на принадлежность множеству;
* 2 запроса с проверкой на принадлежность диапазону значений;
* 2 запроса с проверкой на соответствие шаблону;
* 2 запроса с проверкой на неопределенное значение.

3. Для созданной базы данных, согласно номеру варианта, самостоятельно создать **на языке Transact-SQL многотабличные запросы:**

* 2 запроса с использованием декартового произведения двух таблиц;
* 3 запроса с использованием соединения двух таблиц по равенству;
* 2 запроса с использованием соединения двух таблиц по равенству и условием отбора;
* 2 запроса с использованием соединения по трем таблицам;
* создать копии ранее созданных запросов на соединение по равенству на запросы с использованием внешнего полного соединения таблиц (**JOIN**).
* 2 запроса с использованием левого внешнего соединения;
* 2 запроса на использование правого внешнего соединения;
* 2 запроса с использованием симметричного соединения и удаление избыточности.

1. Для созданной базы данных, согласно номеру варианта, самостоятельно создать на языке Transact-SQL многотабличные запросы:

* 2 запроса с использованием функции COUNT;
* 2 запроса с использованием функции SUM;
* 2 запроса с использованием функций UPPER, LOWER;
* 2 запроса с использованием временных функций;
* 2 запроса с использованием группировки по одному столбцу;
* 2 запроса на использование группировки по нескольким столбцам;
* 2 запроса с использованием условия отбора групп HAVING;
* 2 запроса с использованием фразы HAVING без фразы GROUP BY;
* 2 запроса с использованием сортировки по столбцу;
* 2 запроса на добавление новых данных в таблицу;
* 2 запроса на добавление новых данных по результатам запроса в качестве вставляемого значения;
* 2 запроса на обновление существующих данных в таблице;
* 2 запроса на обновление существующих данных по результатам подзапроса во фразе WHERE;
* 2 запроса на удаление существующих данных.

1. **Выполнение работы**

Часть 1.

CREATE database hospital;

use hospital;

create table patient

( patient\_id int primary key AUTO\_INCREMENT,

disease\_history\_id int not null unique,

patient\_name varchar(45) not null,

surname varchar(90) not null,

patronomyc varchar(90) not null,

adress varchar (90) not null,

city varchar(90) not null,

age int not null

check (age > 0),

sex varchar(10) not null

);

create table doctor

( doctor\_id int primary key auto\_increment,

department\_id int not null auto\_increment,

doctor\_name varchar(45) not null,

surname varchar(90) not null,

patronomyc varchar(90) not null,

job\_title varchar(50) not null,

expirience int

check (expirience > 0),

doctor\_rank varchar(100),

address varchar(100)

);

create table disease\_history

( disease\_history\_id int primary key AUTO\_INCREMENT,

treatment varchar(100) not null,

date\_of\_illness date not null,

date\_of\_cure date not null,

type\_of\_treatment varchar(150) not null

);

create table dignosis

( diagnosis\_id int primary key auto\_increment,

diagnosis\_name varchar(100) not null,

signs\_of\_illness varchar(150) not null,

treatment\_period int not null,

destination varchar(100)

);

create table department

( department\_id int primary key auto\_increment,

department\_name varchar(50) not null,

stage int not null,

room\_number int

check (room\_number > 0),

manager varchar(100) not null

);

create table diagnosis\_disease\_history

( diagnosis\_id int,

disease\_history\_id int,

primary key (diagnosis\_id,disease\_history\_id)

);

create table doctor\_disease\_history

( doctor\_id int,

disease\_history\_id int,

primary key(doctor\_id,disease\_history\_id)

);

create table department\_patient

( department\_id int,

patient\_id int,

primary key(department\_id,patient\_id)

);

alter table diagnosis\_disease\_history

add constraint diagnosis\_disease\_history\_fk1

foreign key (diagnosis\_id) references diagnosis(diagnosis\_id);

alter table diagnosis\_disease\_history

add constraint diagnosis\_disease\_history\_fk2

foreign key (disease\_history\_id) references disease\_history(disease\_history\_id);

alter table doctor\_disease\_history

add constraint doctor\_disease\_history\_fk1

foreign key (doctor\_id) references doctor(doctor\_id);

alter table doctor\_disease\_history

add constraint doctor\_disease\_history\_fk2

foreign key (disease\_history\_id) references disease\_history(disease\_history\_id);

alter table department\_patient

add constraint department\_patient\_fk1

foreign key (department\_id) references department(department\_id);

alter table department\_patient

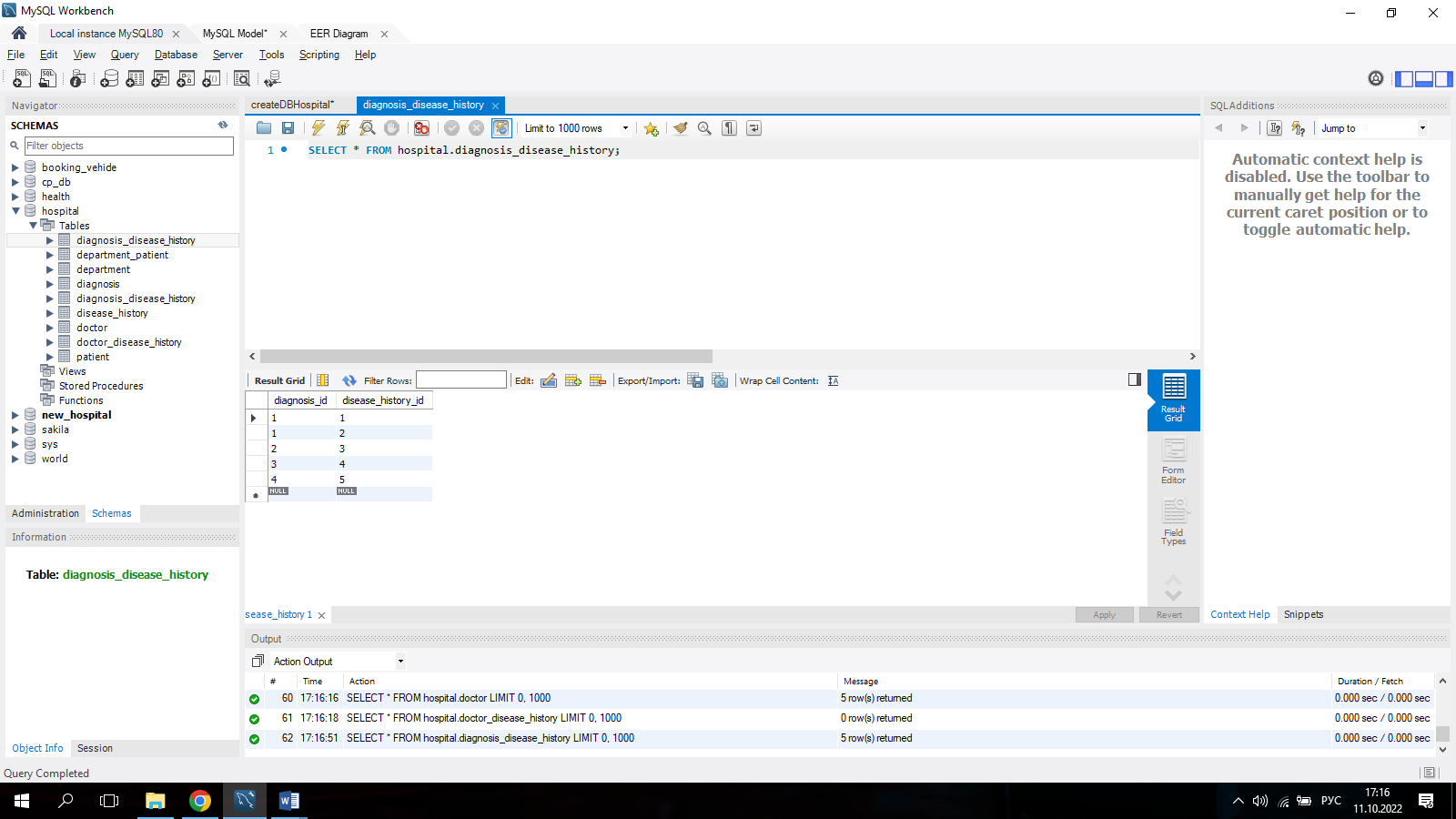
add constraint department\_patient\_fk2

foreign key (patient\_id) references patient(patient\_id);

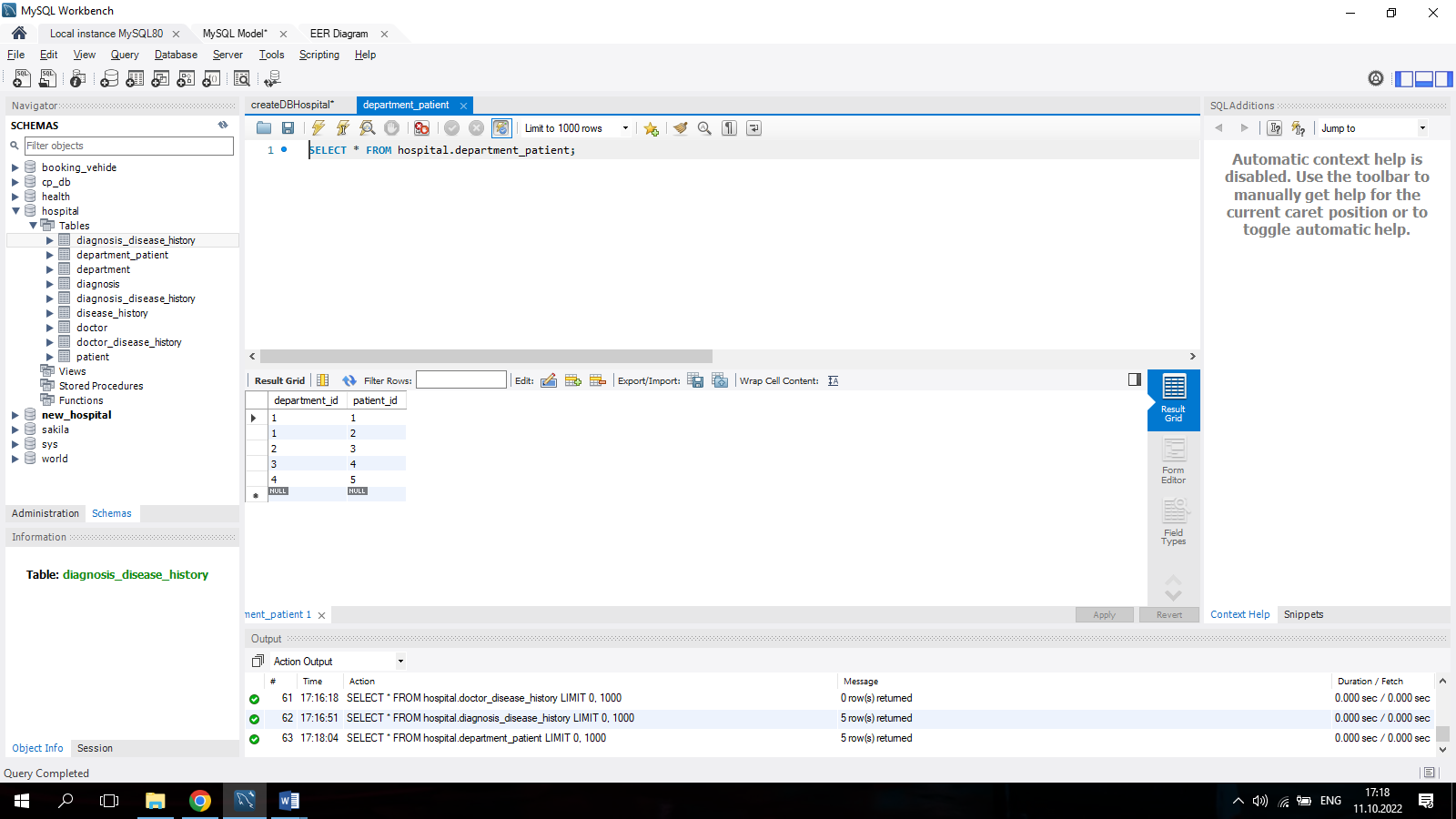
alter table patient add constraint patient\_fk1 foreign key (disease\_history\_id) references disease\_history(disease\_history\_id);

alter table doctor add constraint doctor\_fk1 foreign key (department\_id) references department(department\_id);

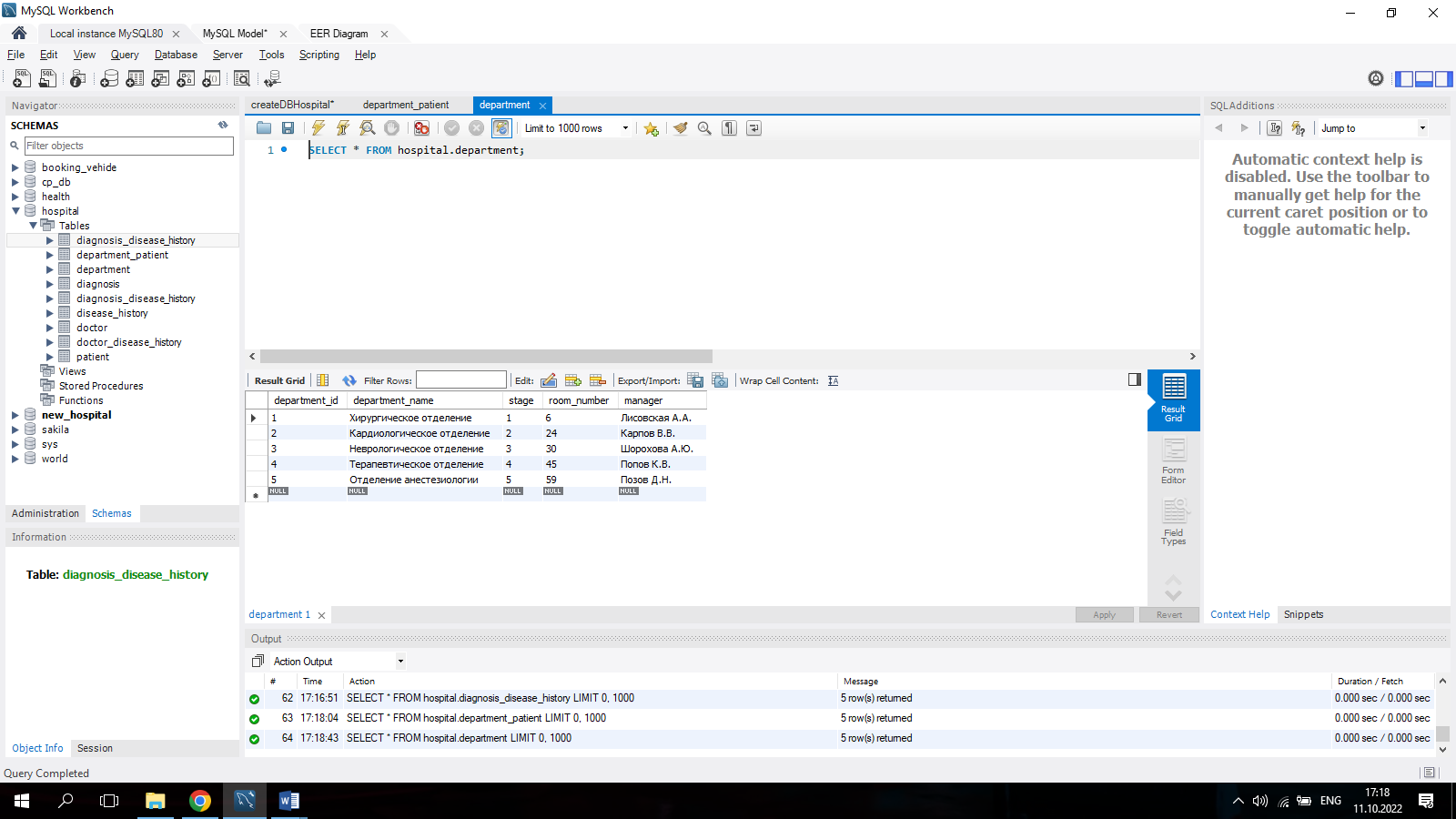
1. Diagnosis\_disease\_history



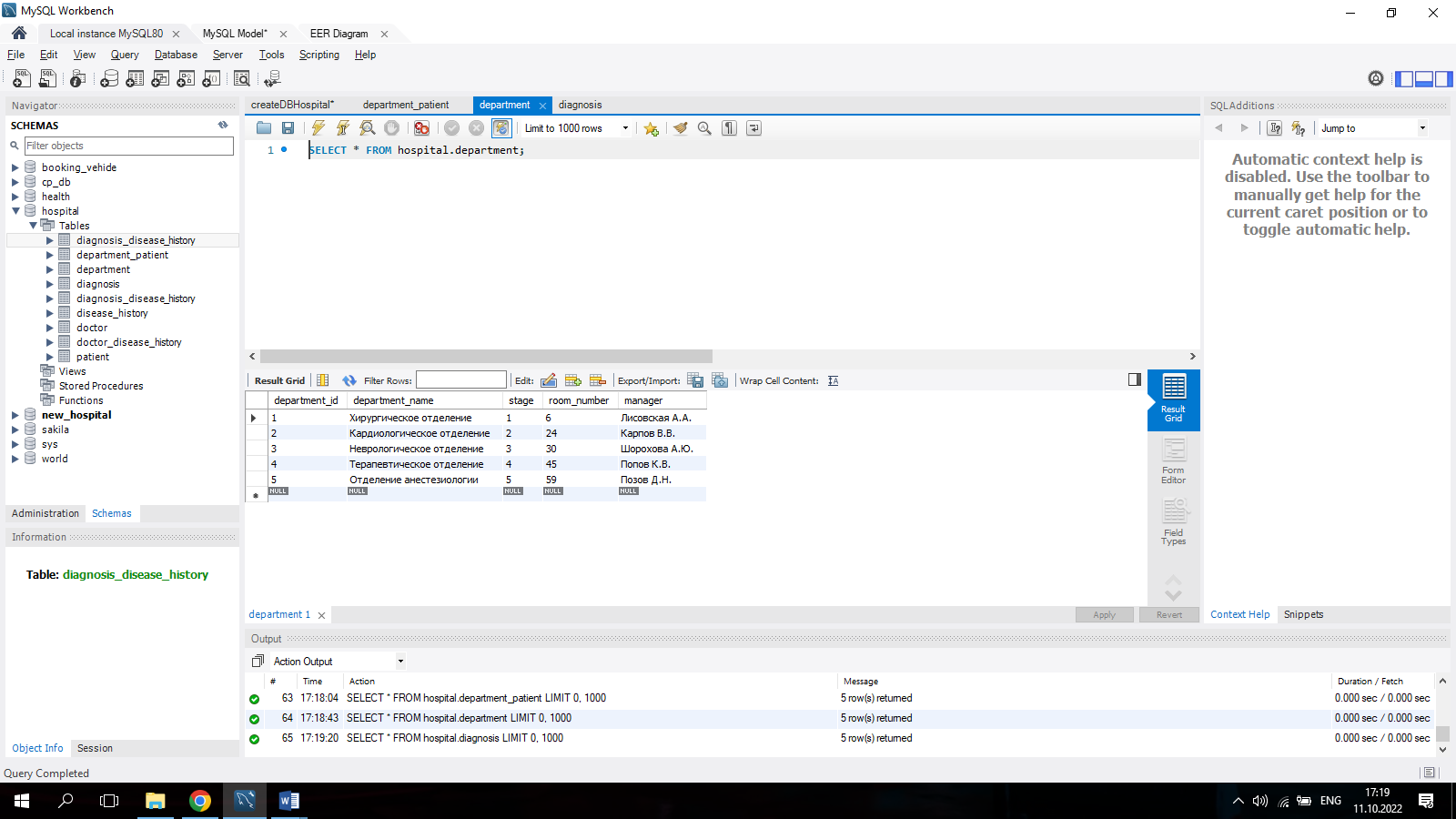
1. Department\_patient



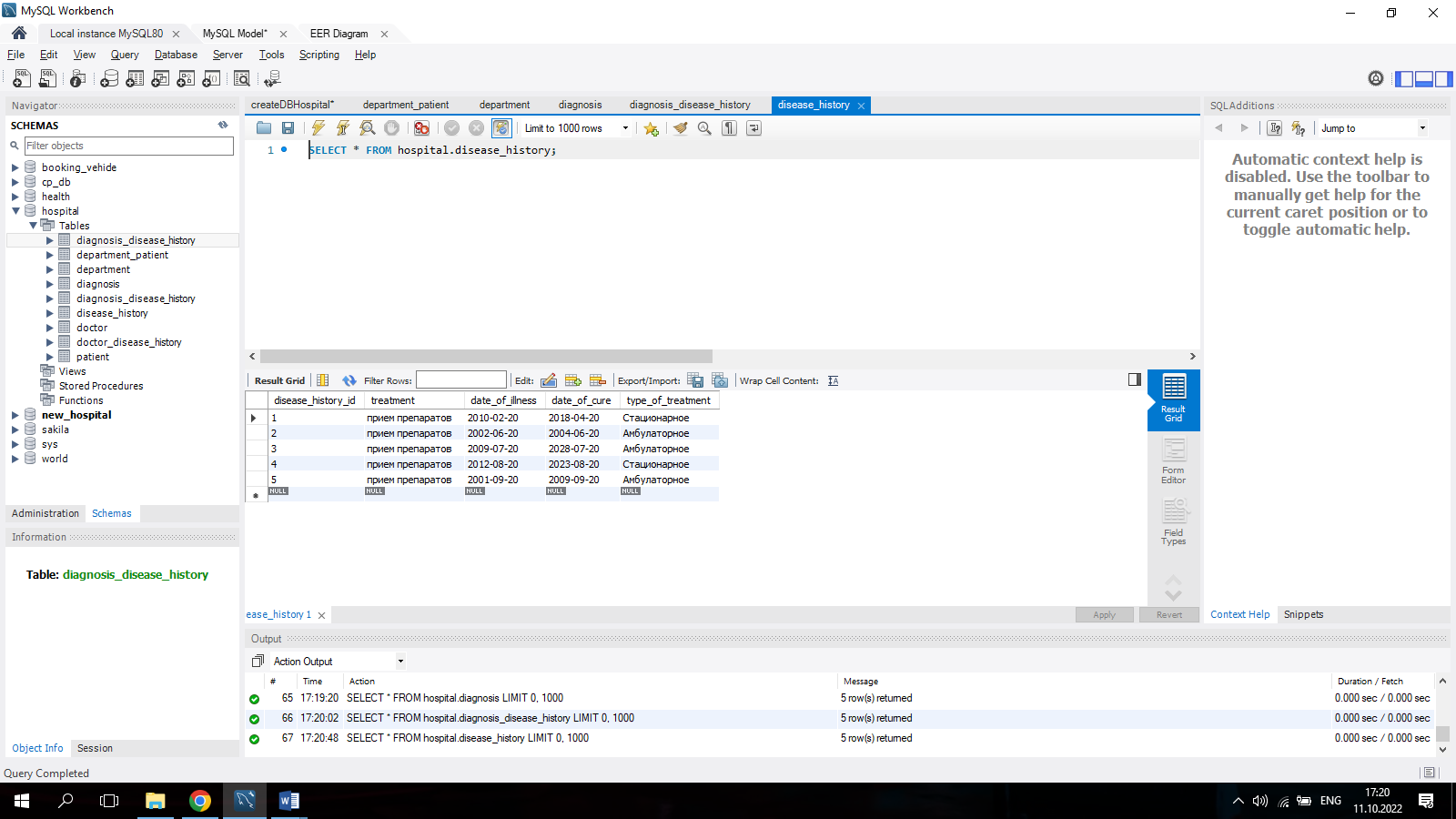
1. Department



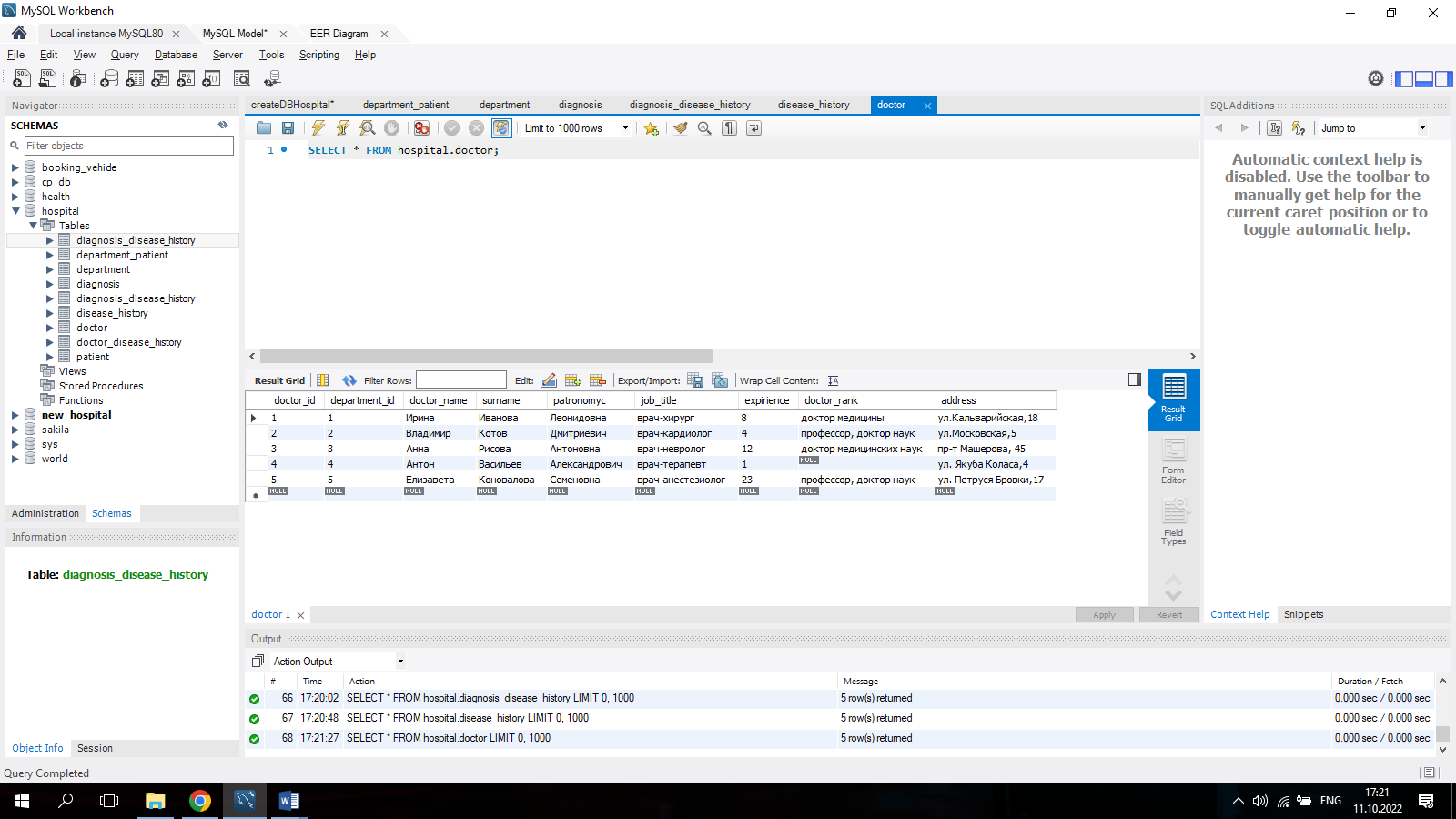
1. Diagnosis



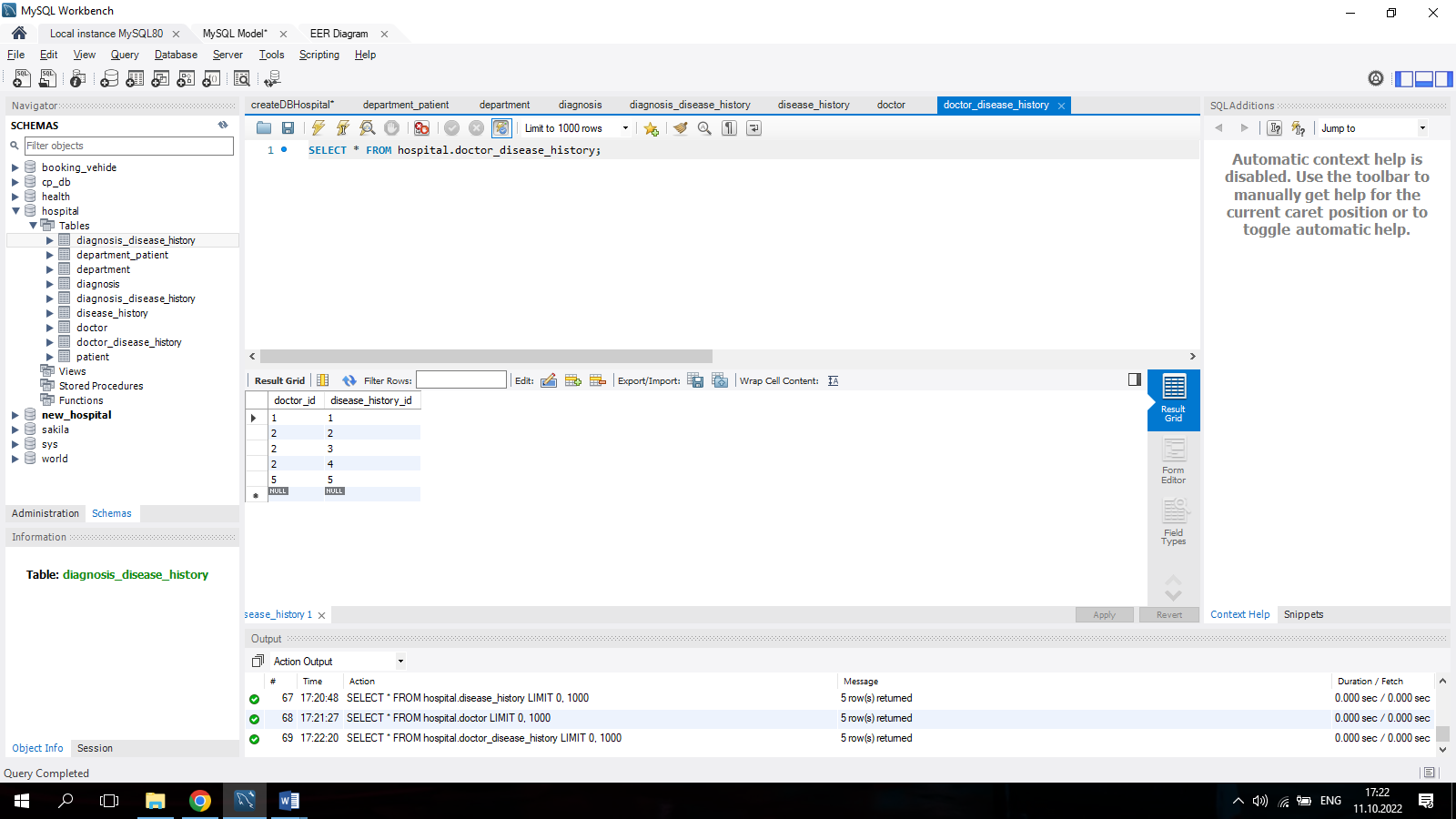
1. Disease\_history



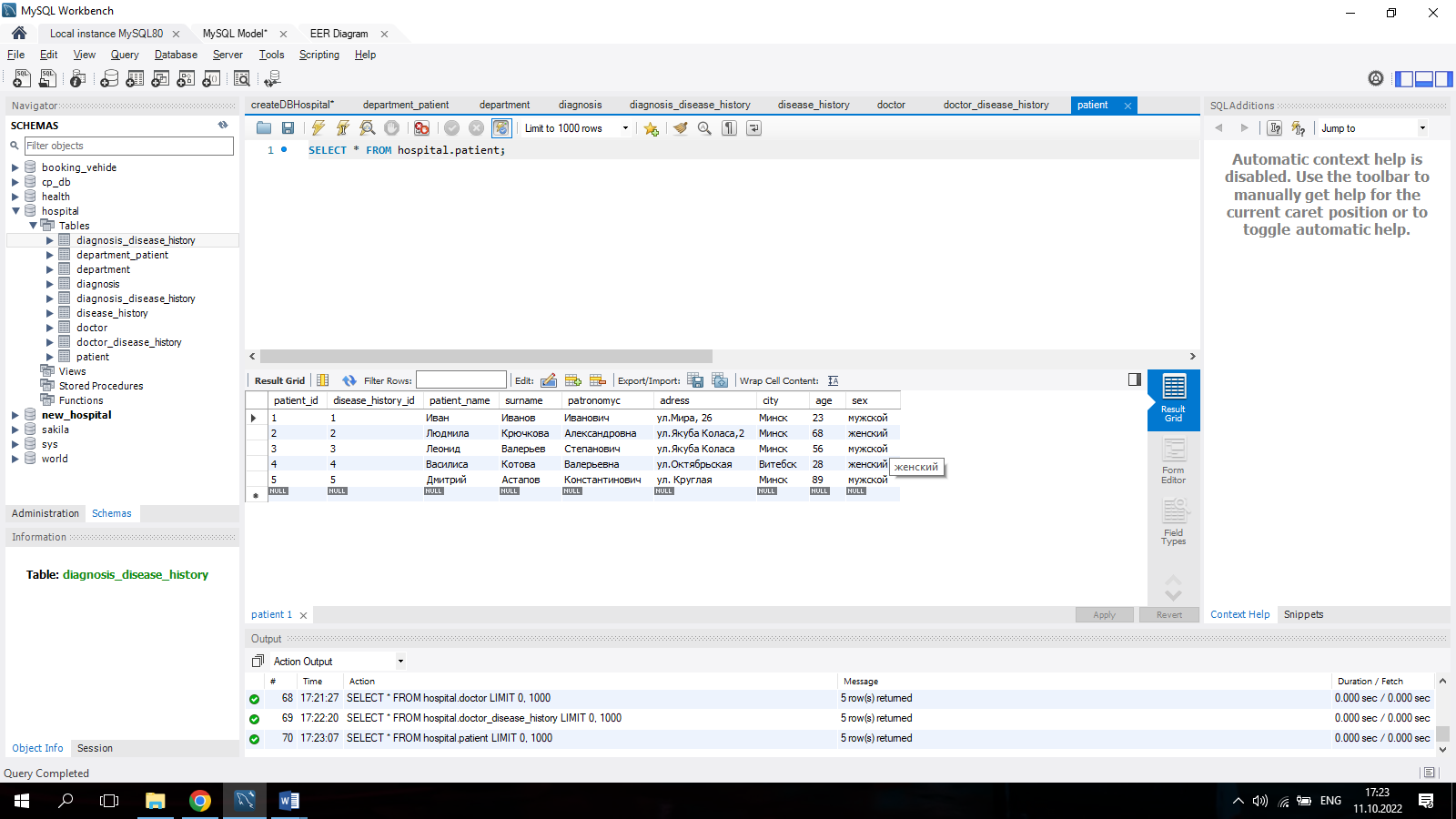
1. Doctor



1. Doctor\_disease\_history



1. Patient

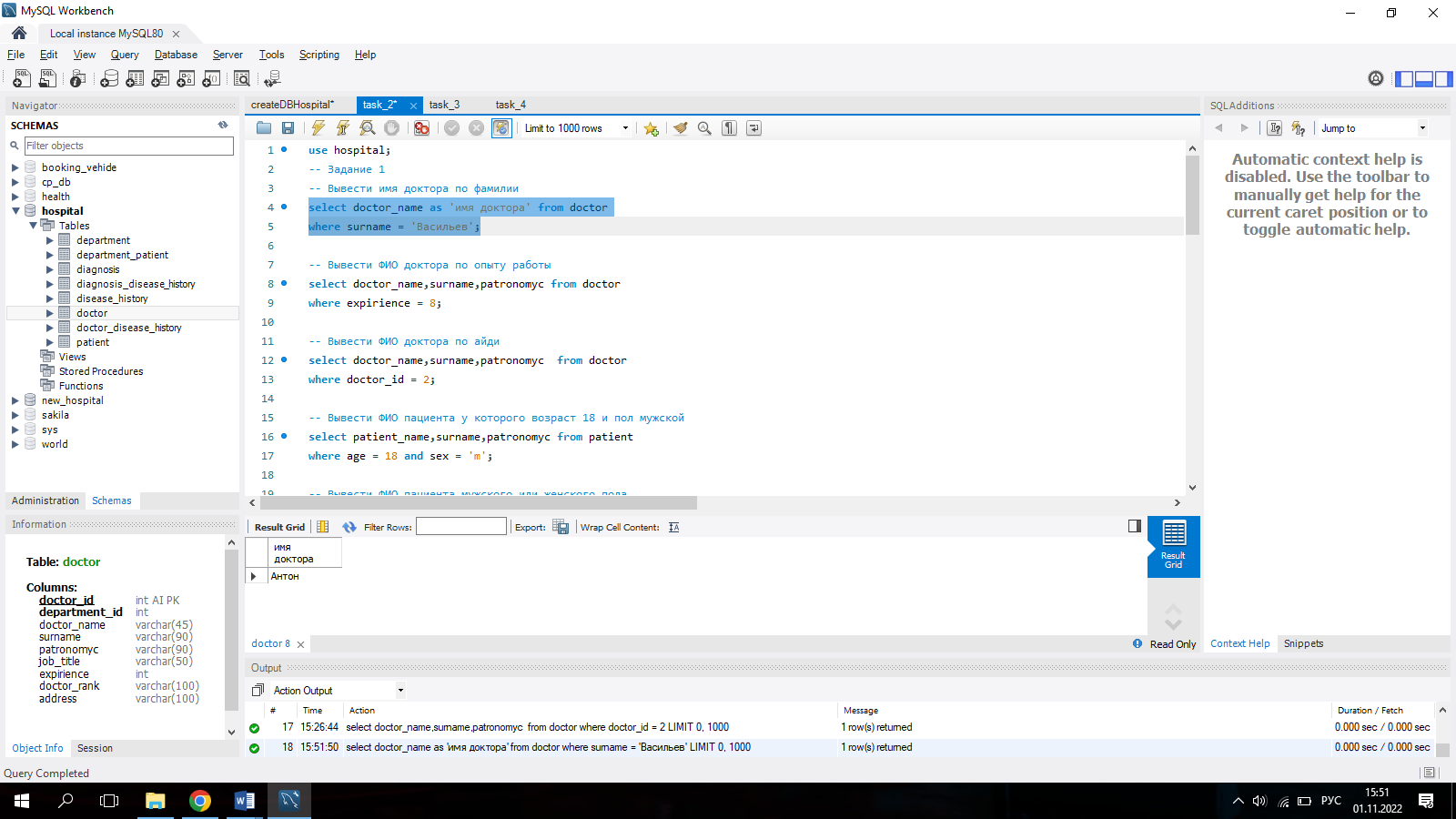


Часть 2.

-- Вывести имя доктора по фамилии

select doctor\_name as 'имя доктора' from doctor

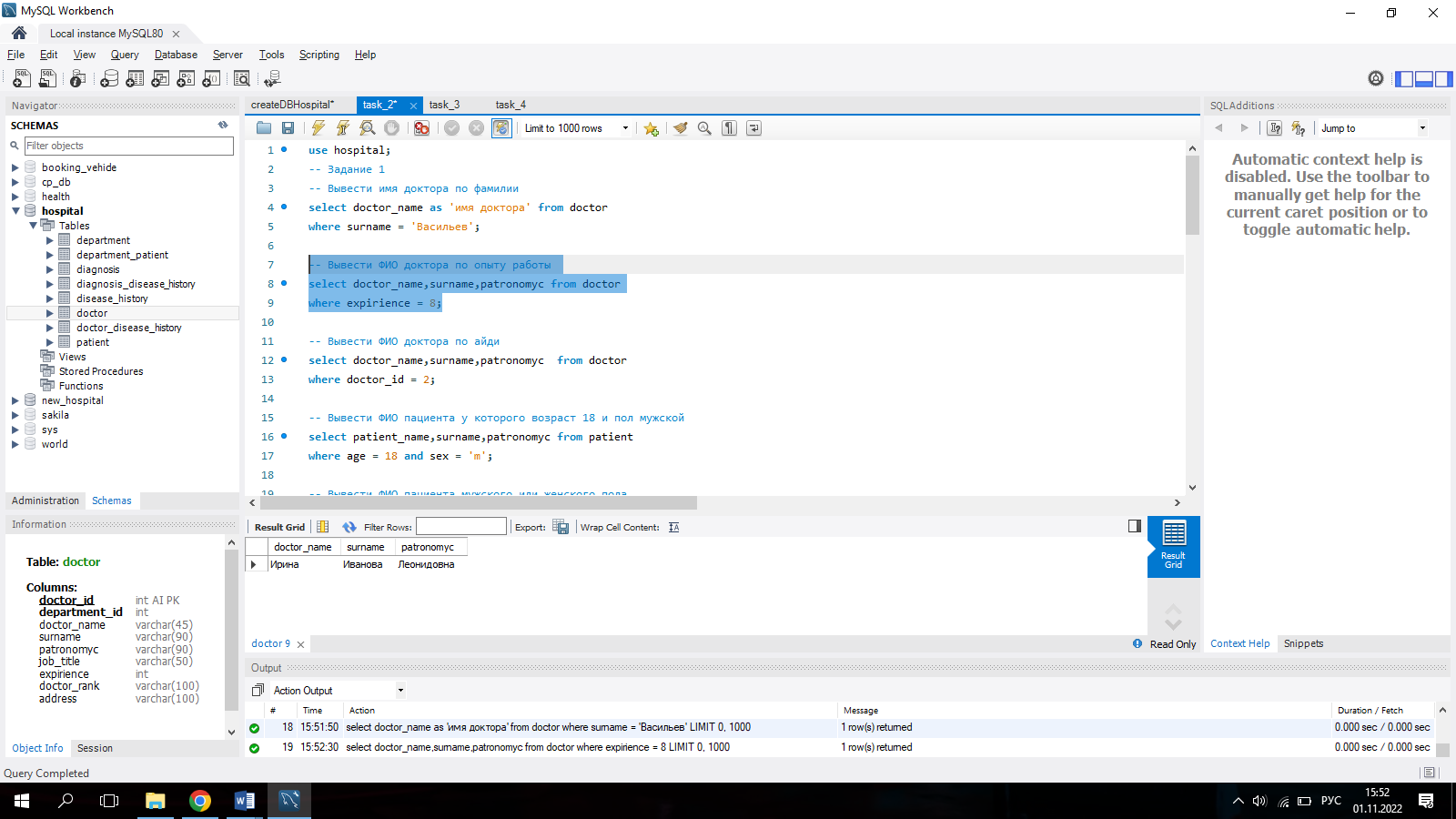
where surname = 'Васильев';



-- Вывести ФИО доктора по опыту работы

select doctor\_name,surname,patronomyc from doctor

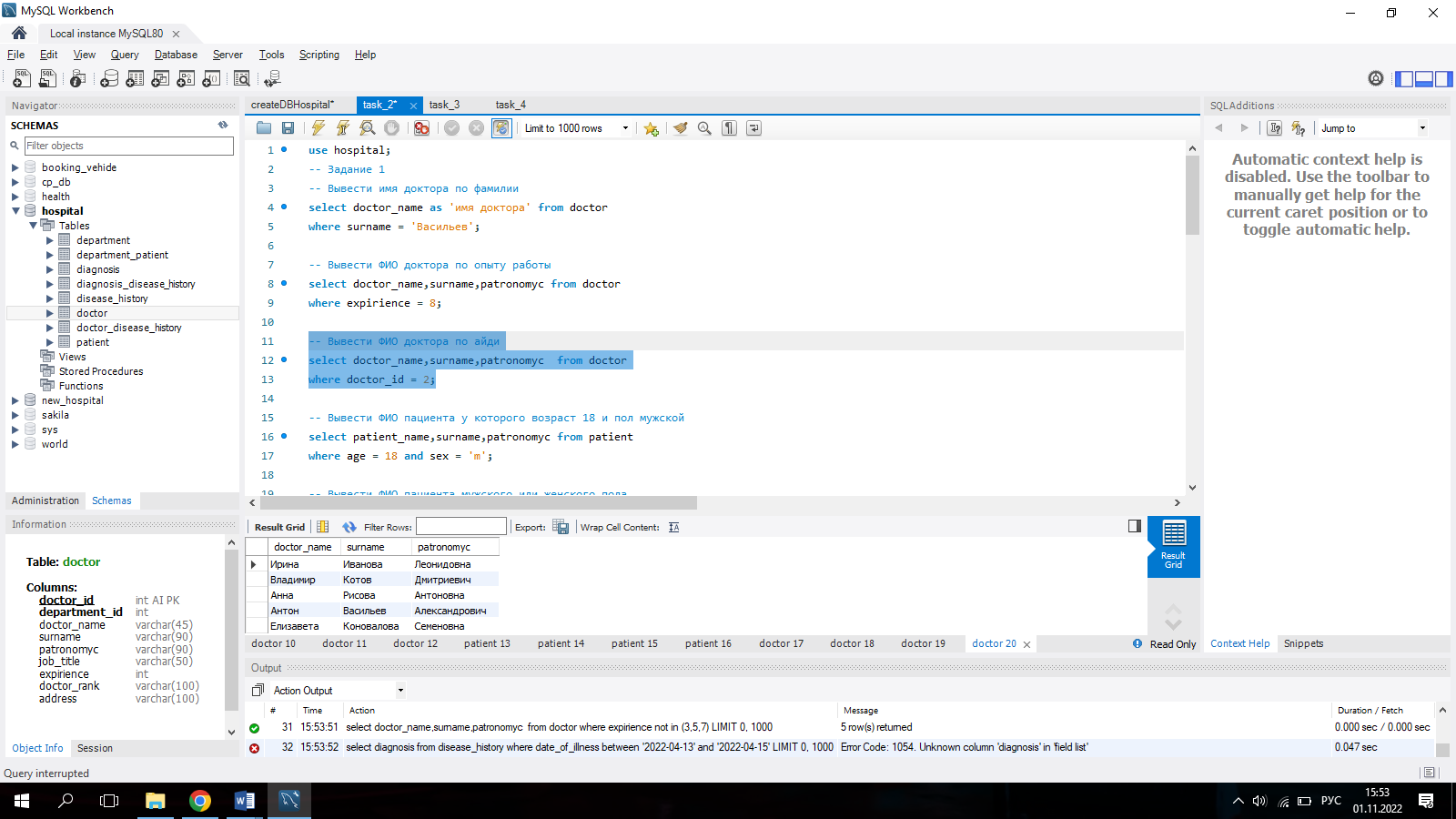
where expirience = 8;



-- Вывести ФИО доктора по айди

select doctor\_name,surname,patronomyc from doctor

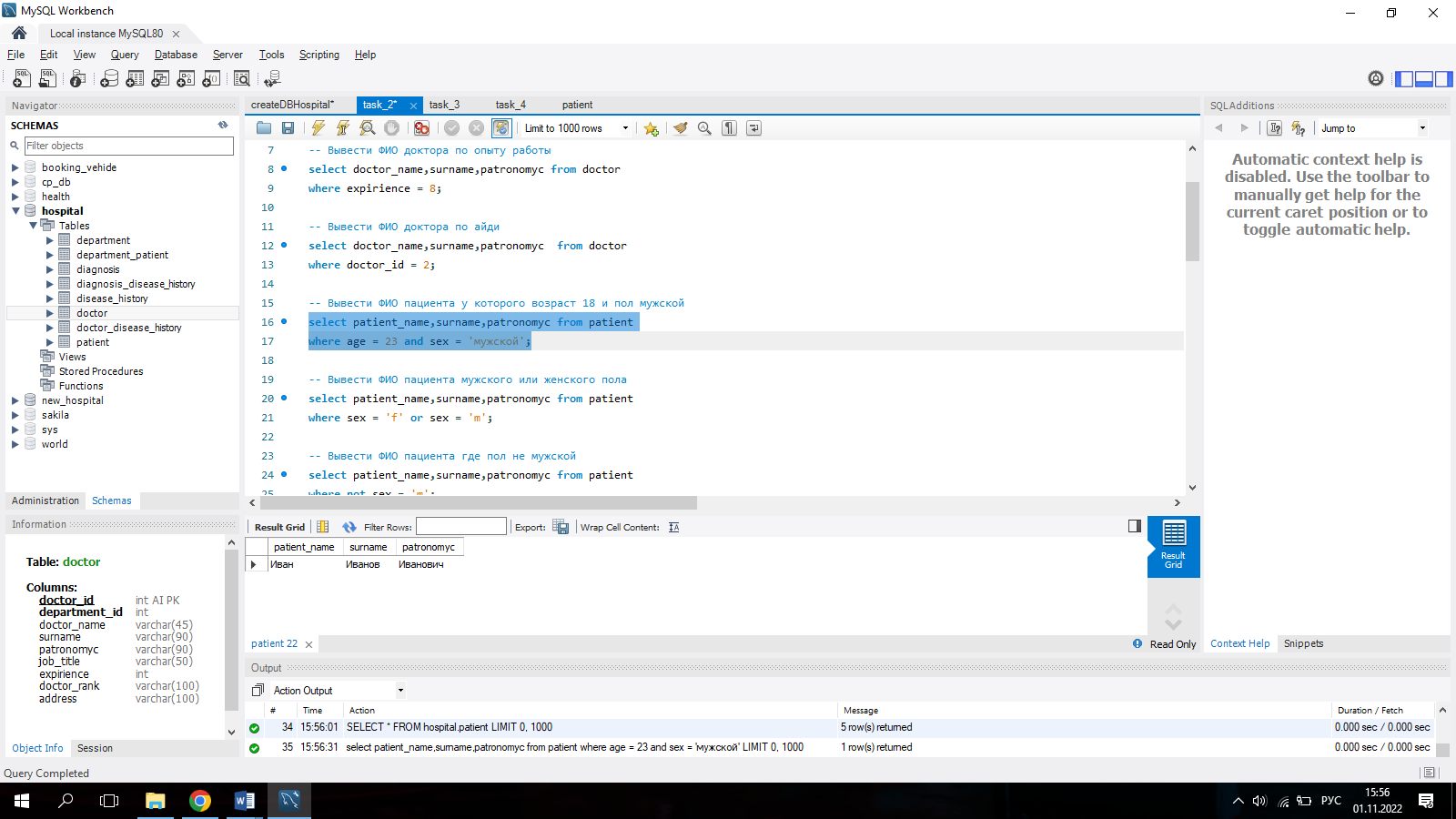
where doctor\_id = 2;



-- Вывести ФИО пациента у которого возраст 18 и пол мужской

select patient\_name,surname,patronomyc from patient

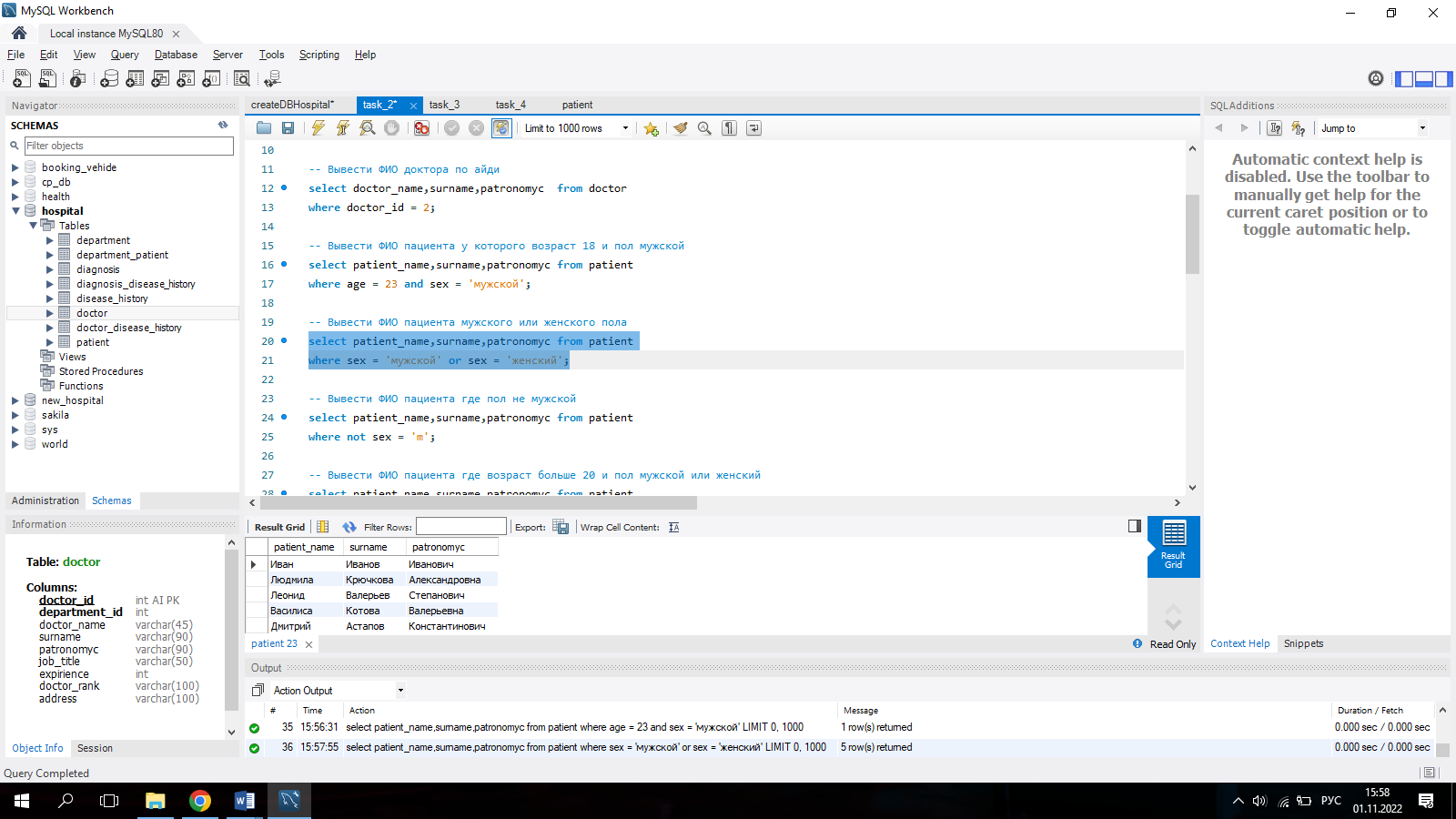
where age = 23 and sex = 'мужской';



-- Вывести ФИО пациента мужского или женского пола

select patient\_name,surname,patronomyc from patient

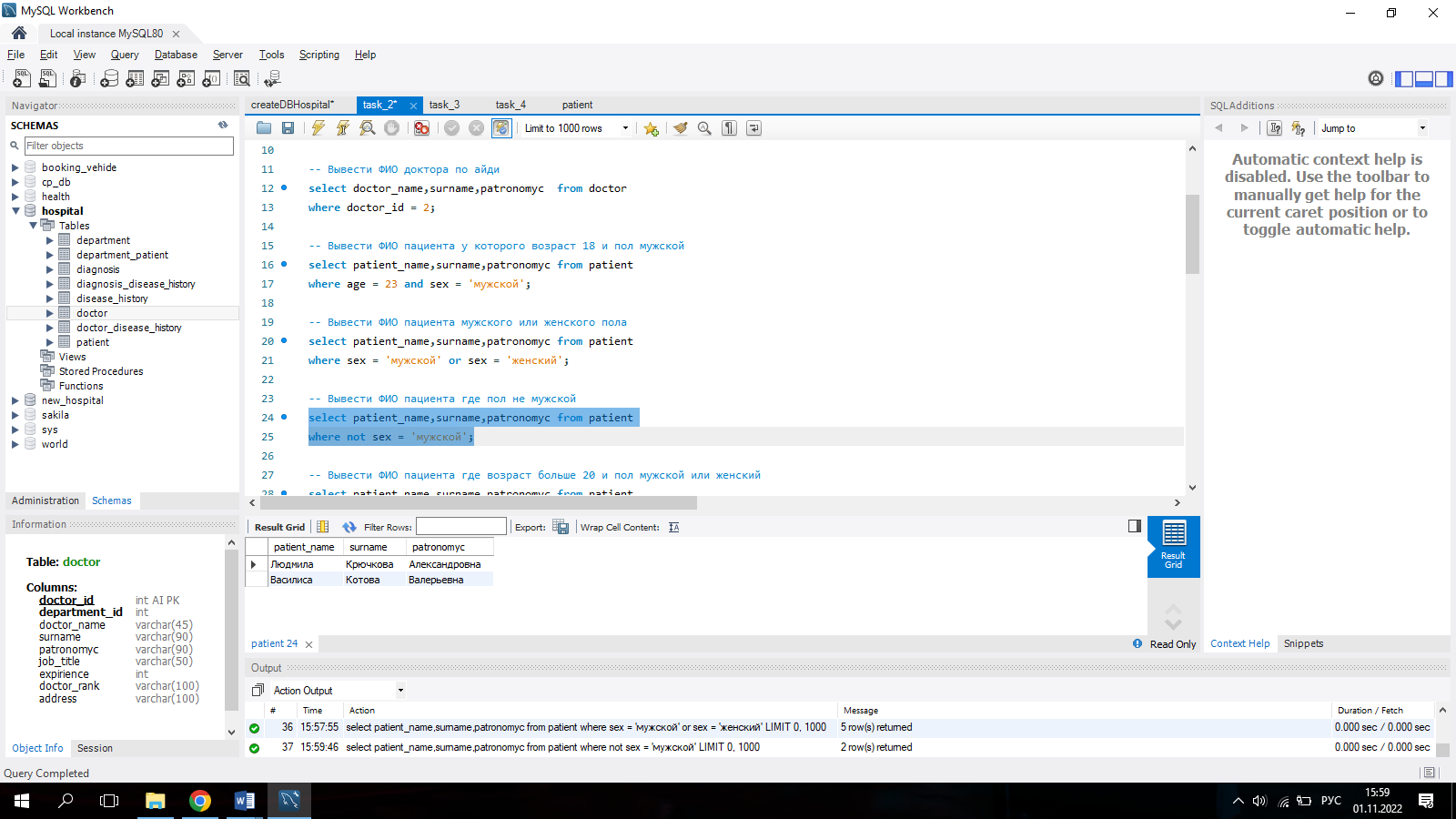
where sex = 'мужской' or sex = 'женский';



-- Вывести ФИО пациента где пол не мужской

select patient\_name,surname,patronomyc from patient

where not sex = 'женский';

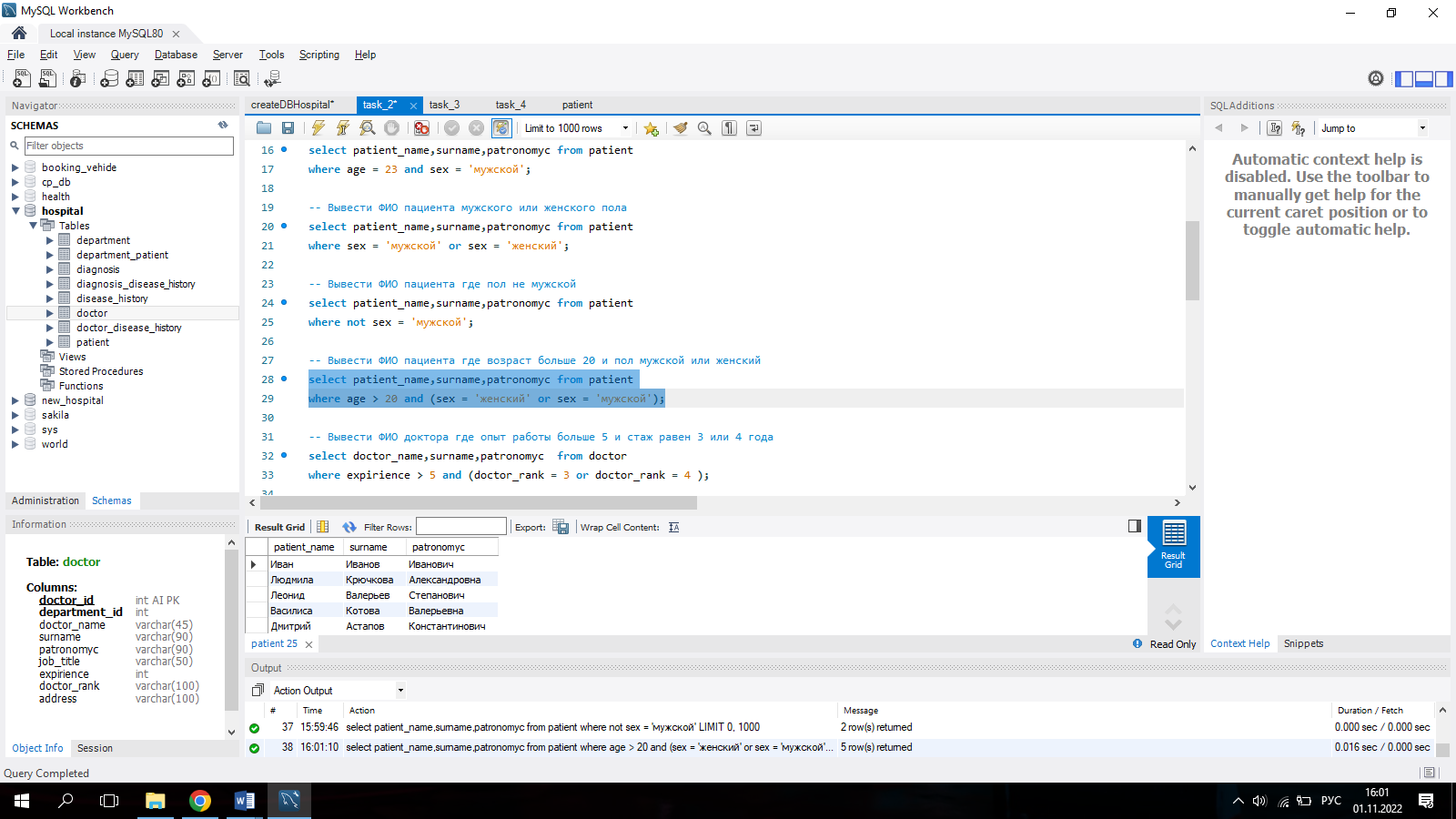




-- Вывести ФИО пациента где возраст больше 20 и пол мужской или женский

select patient\_name,surname,patronomyc from patient

where age > 20 and (sex = 'женский' or sex = 'мужской');

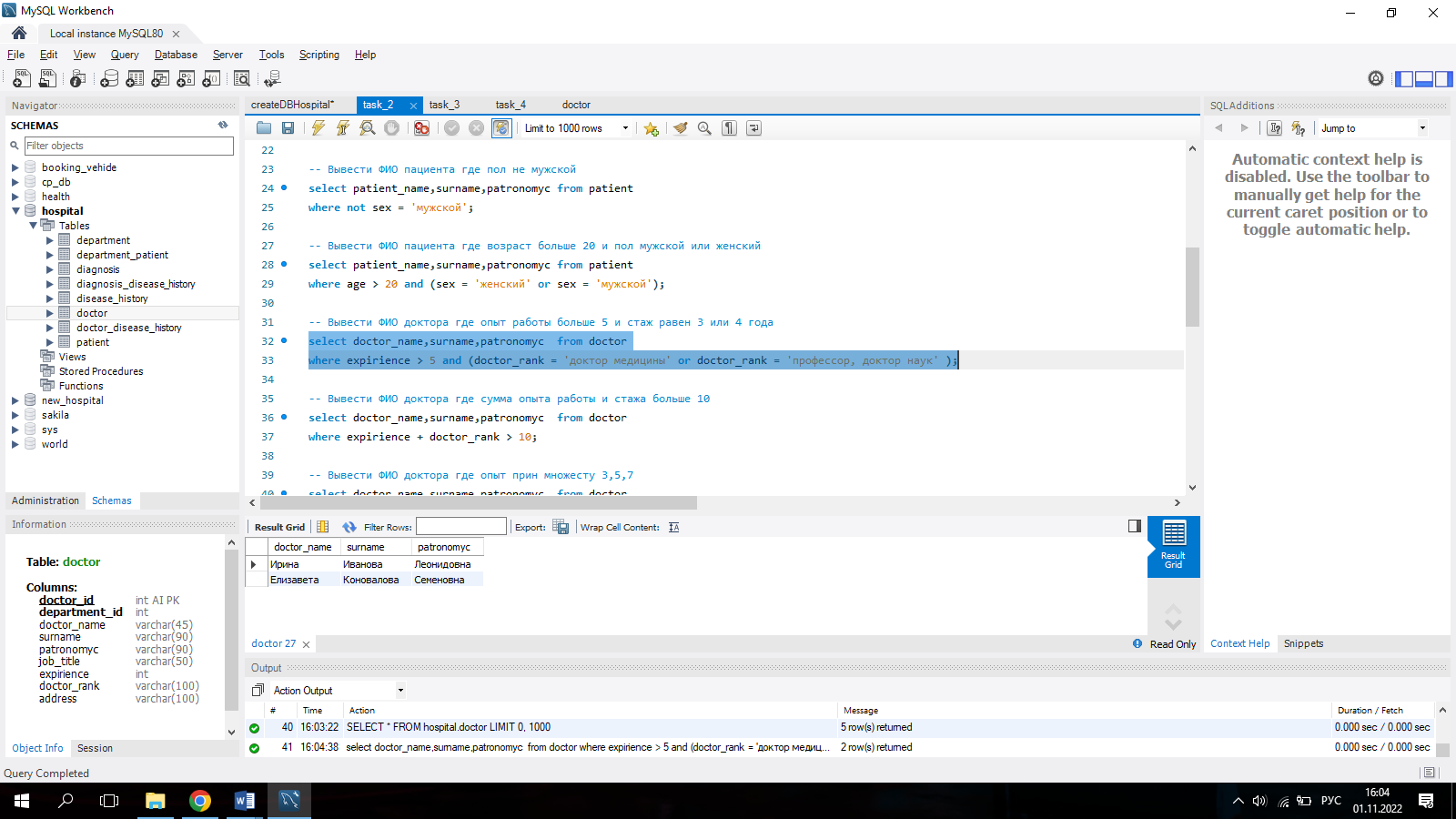




-- Вывести ФИО доктора где опыт работы больше 5 и ранг профессор, доктор наук или доктор медицины

select doctor\_name,surname,patronomyc from doctor

where expirience > 5 and (doctor\_rank = 'доктор медицины' or doctor\_rank = 'профессор, доктор наук' );

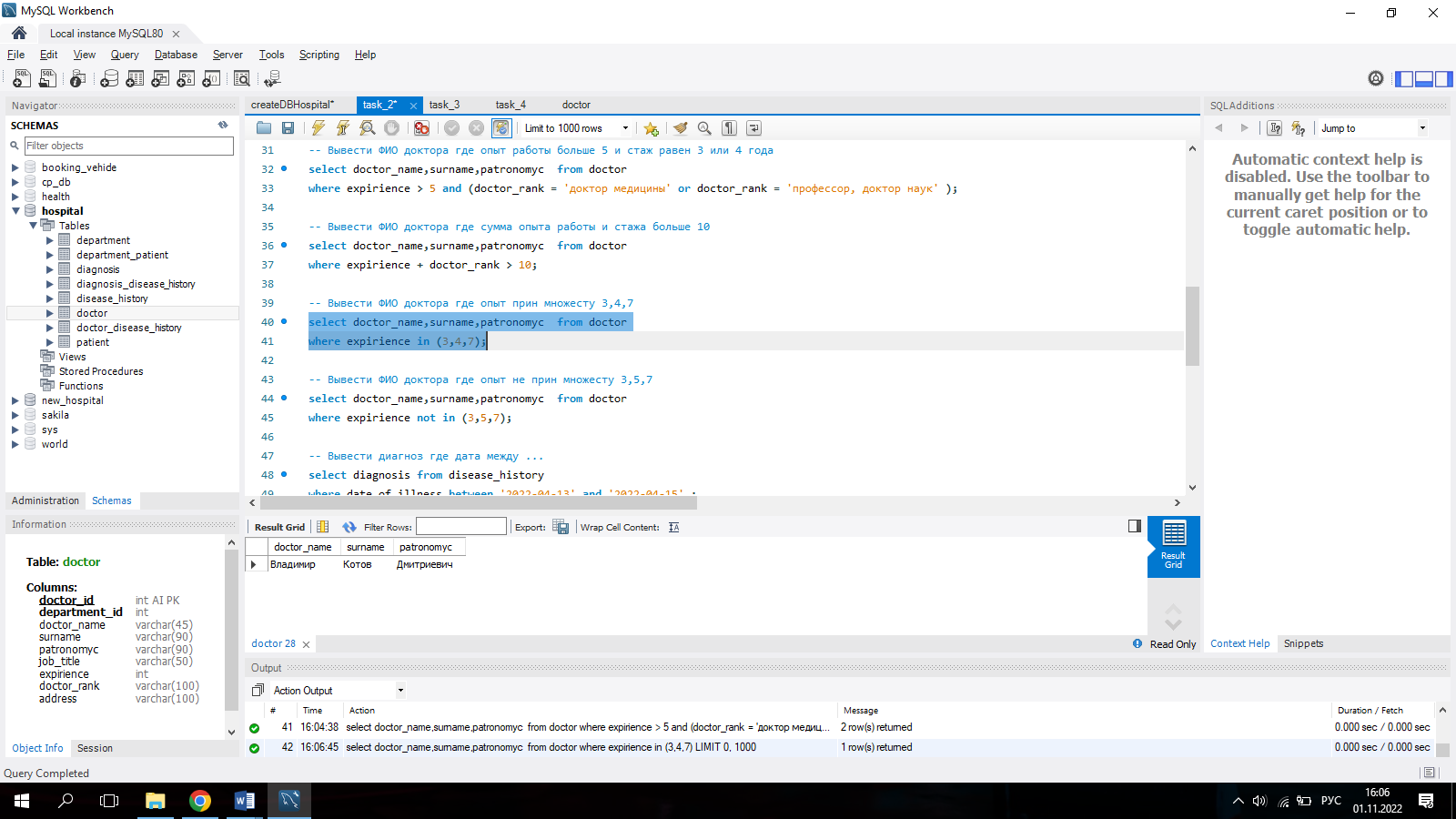




-- Вывести ФИО доктора где опыт прин множесту 3,4,7

select doctor\_name,surname,patronomyc from doctor

where expirience in (3,4,7);

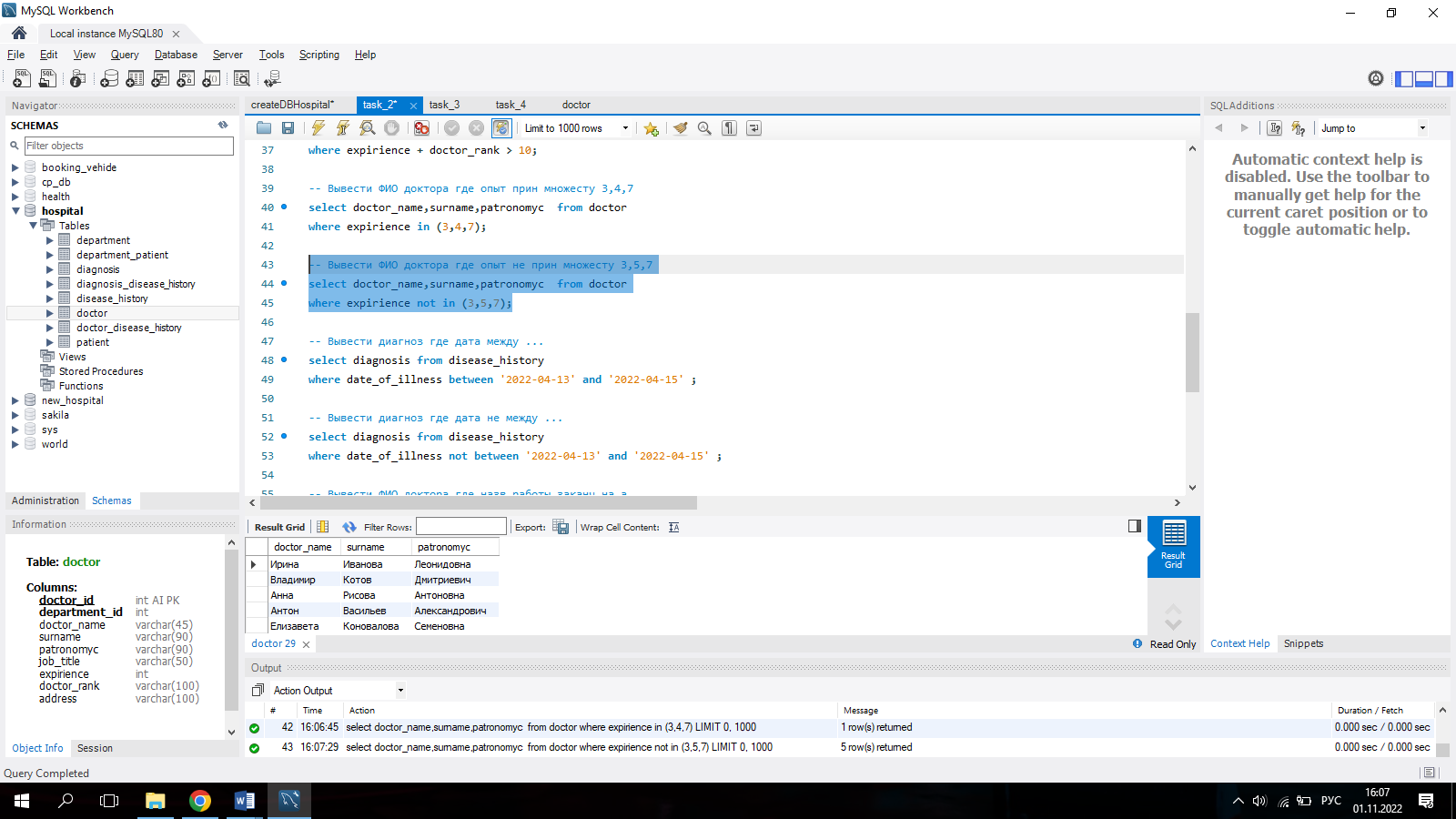




-- Вывести ФИО доктора где опыт не прин множесту 3,5,7

select doctor\_name,surname,patronomyc from doctor

where expirience not in (3,5,7);





-- Вывести тип лечения где дата между ...

select type\_of\_treatment from disease\_history

where date\_of\_illness between '2010-02-16' and '2010-02-25' ;

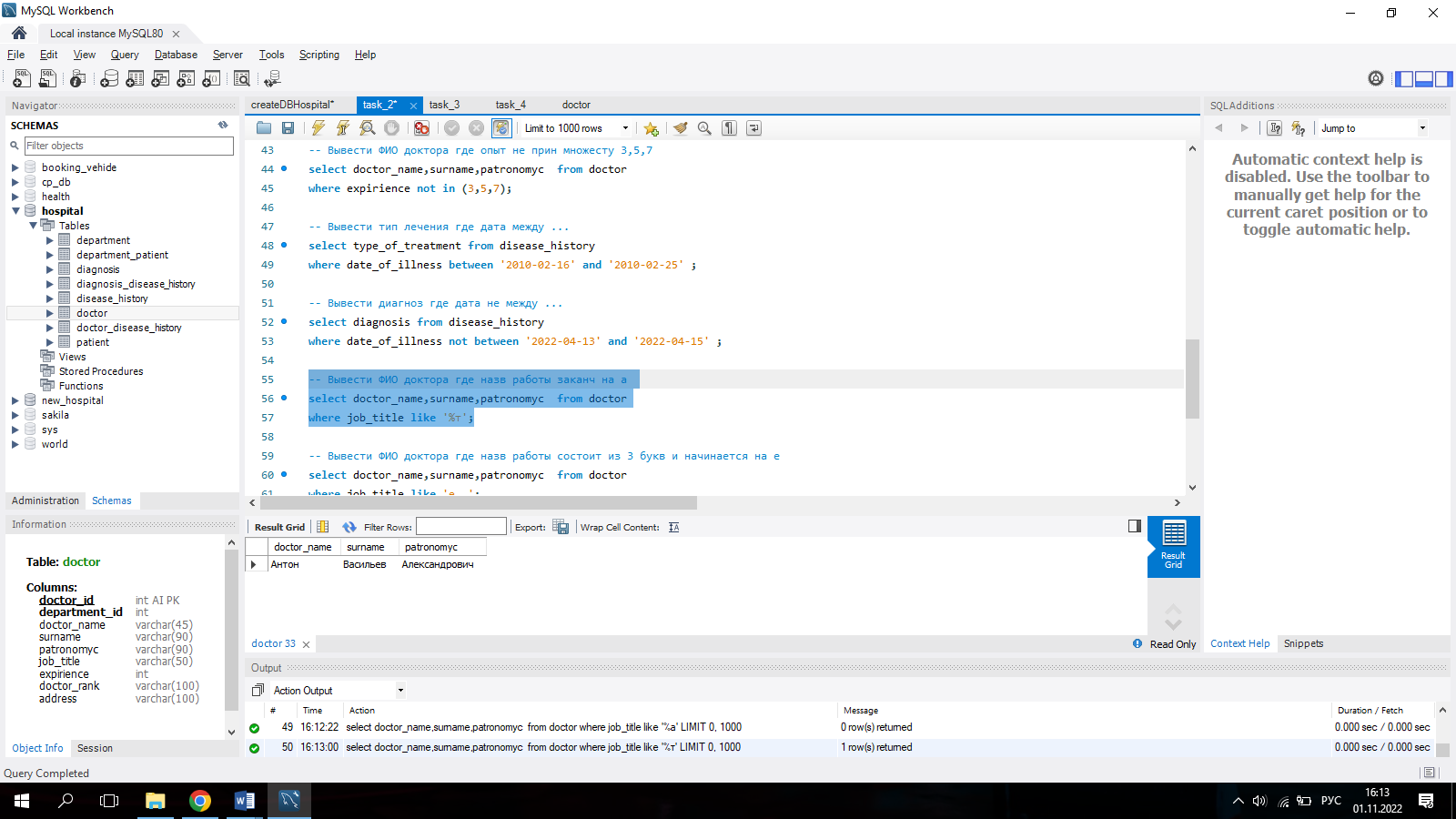




-- Вывести ФИО доктора где назв работы заканч на т

select doctor\_name,surname,patronomyc from doctor

where job\_title like '%т';

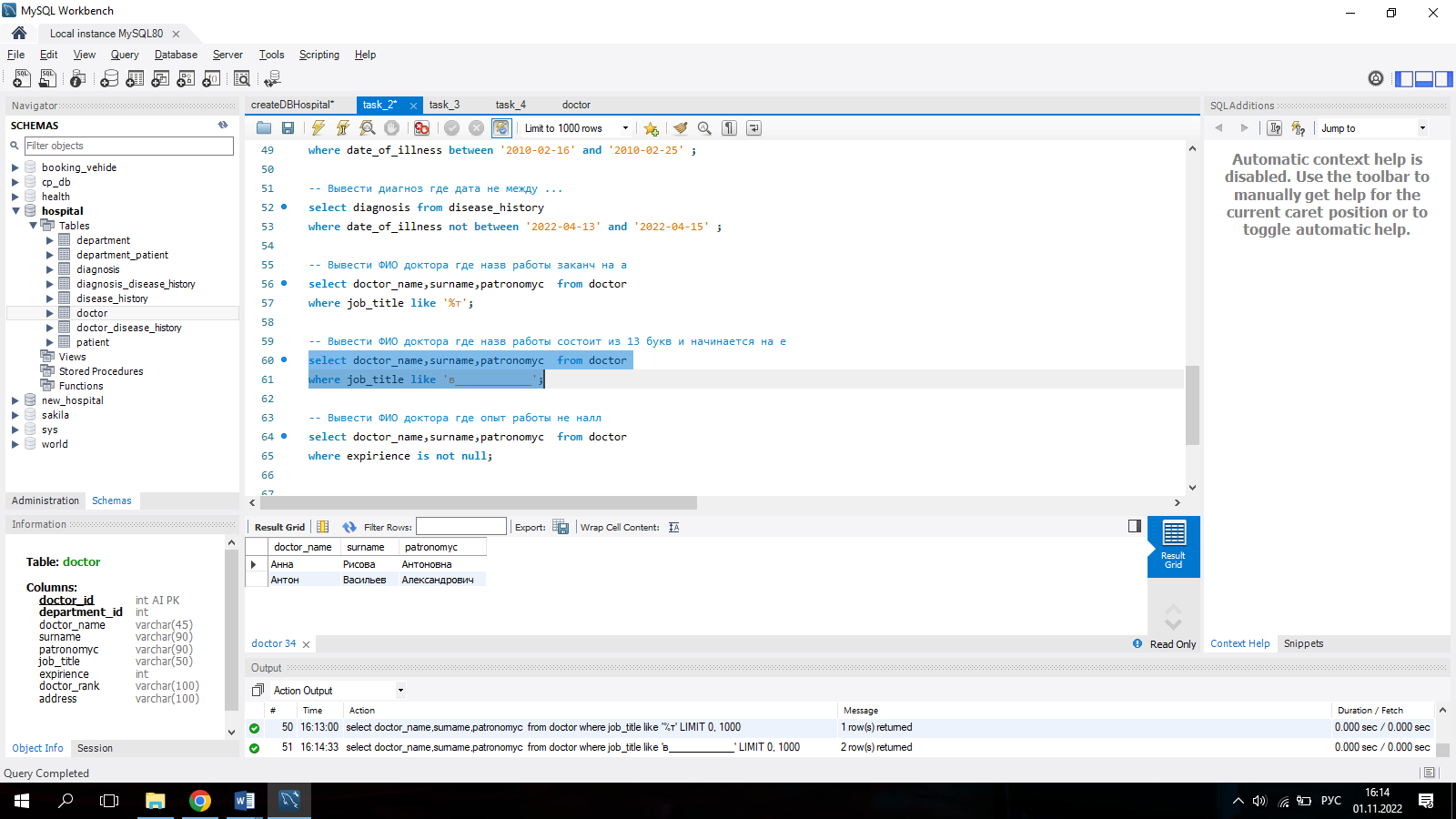




-- Вывести ФИО доктора где назв работы состоит из 13 букв и начинается на е

select doctor\_name,surname,patronomyc from doctor

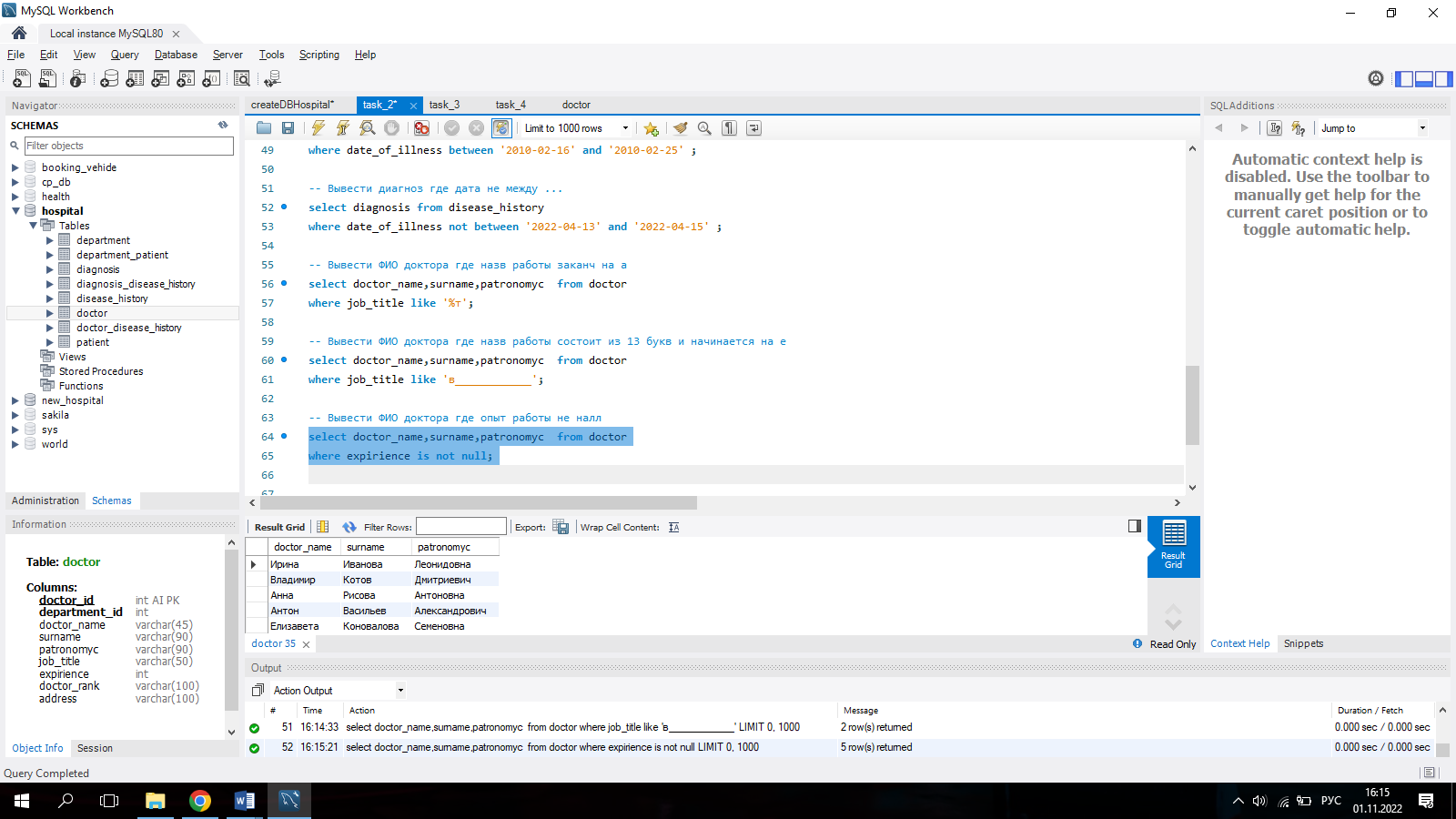
where job\_title like 'в\_\_\_\_\_\_\_\_\_\_\_\_';



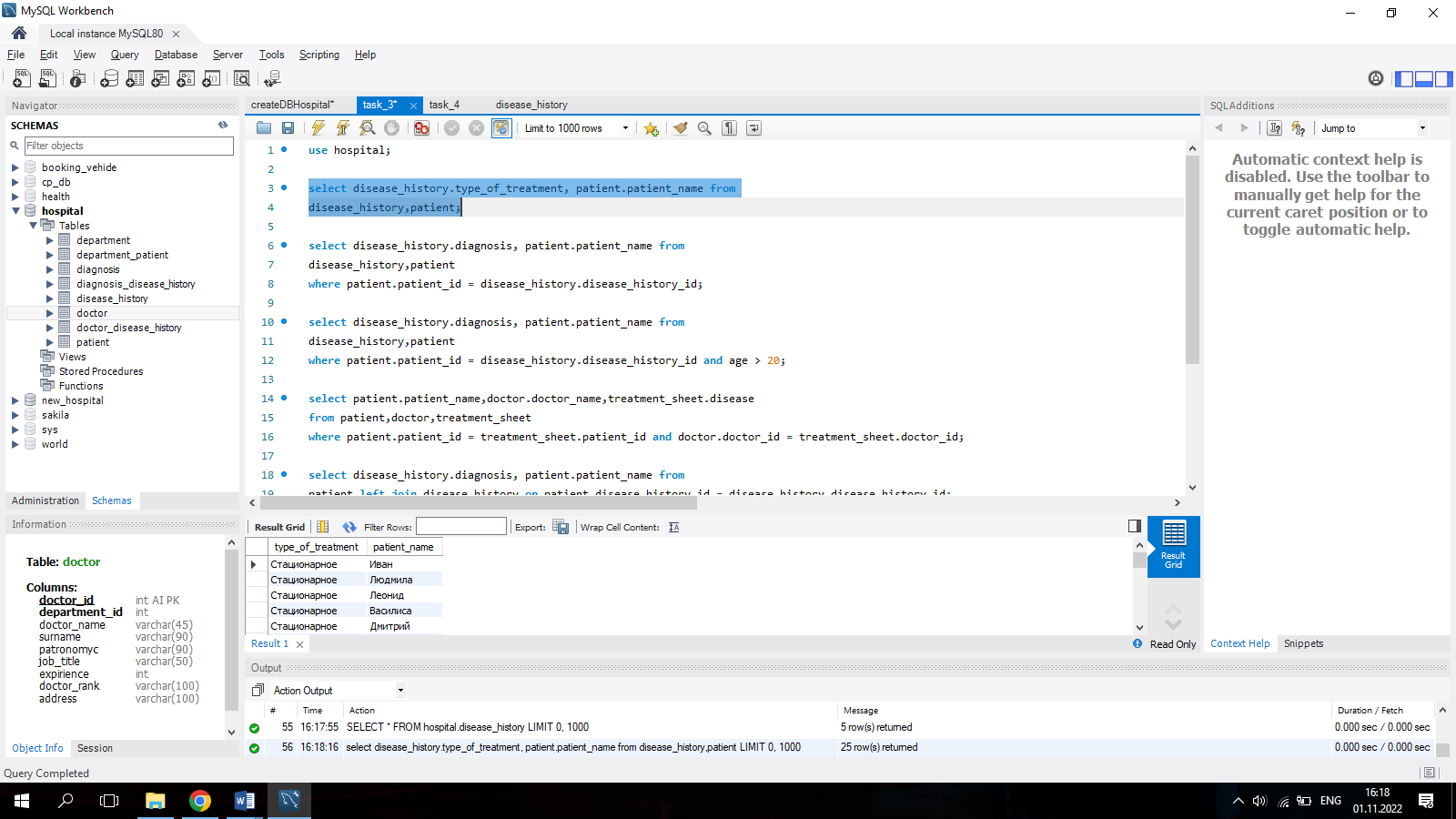
-- Вывести ФИО доктора где опыт работы не налл

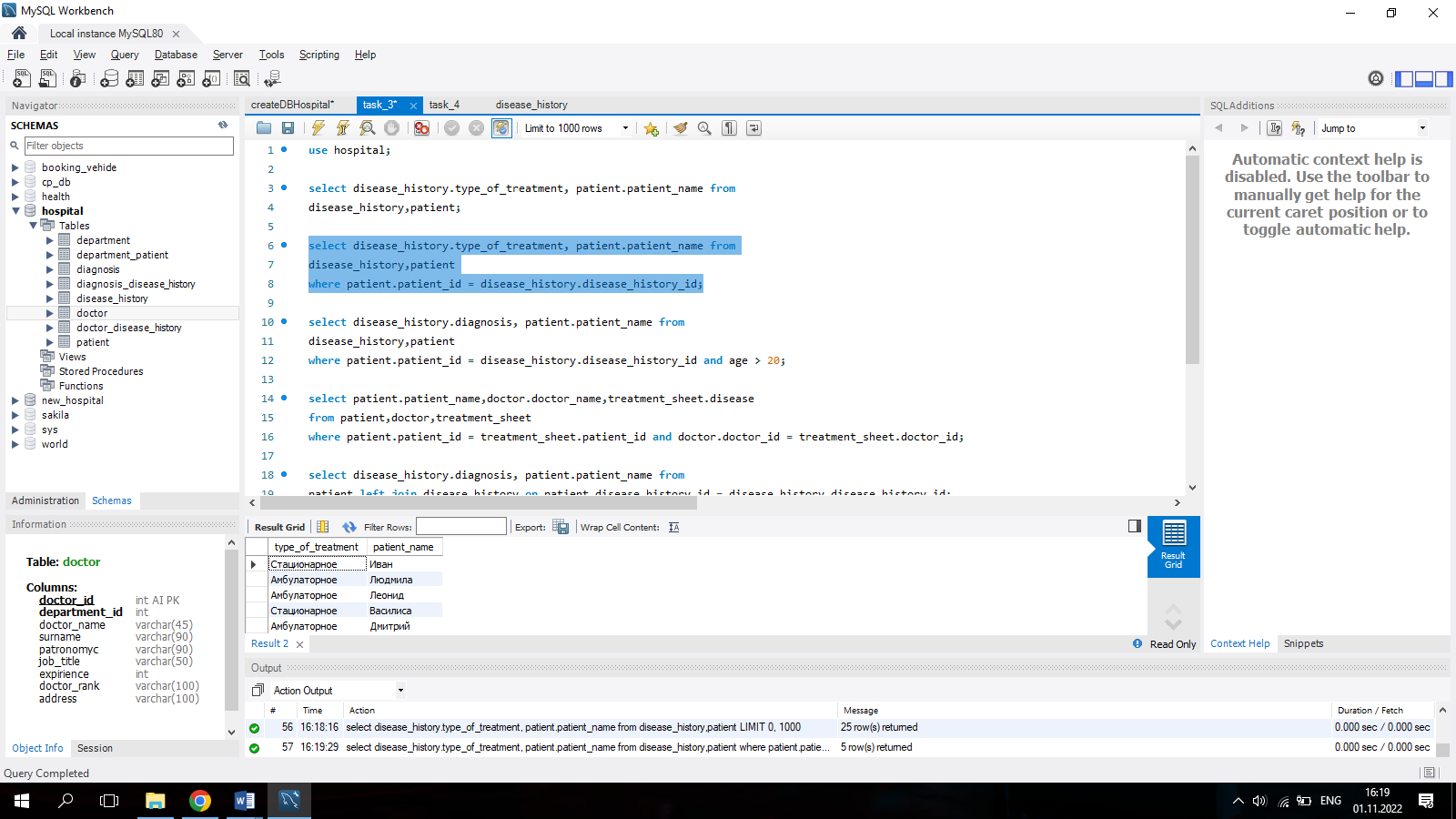
select doctor\_name,surname,patronomyc from doctor

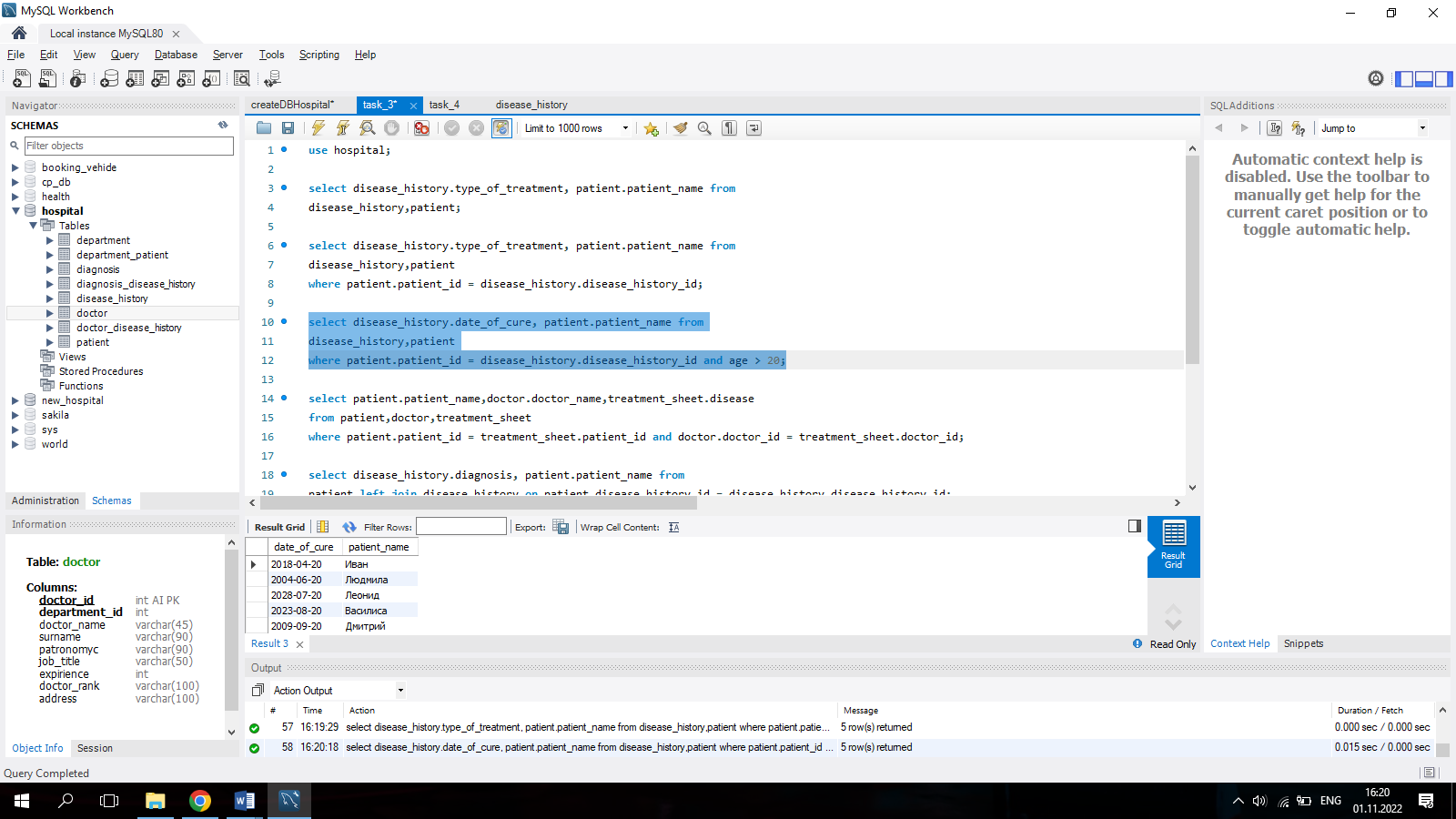
where expirience is not null;

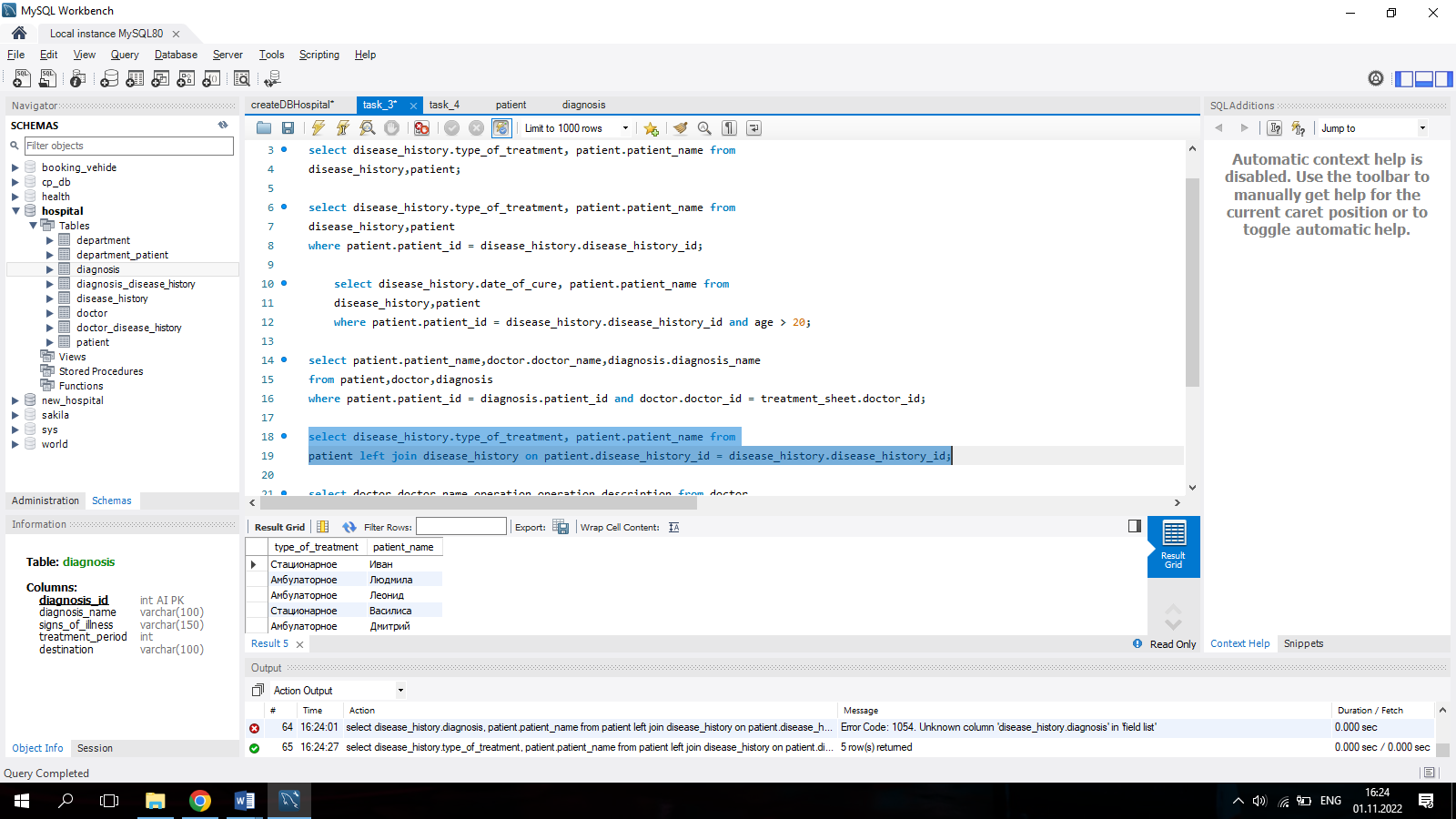


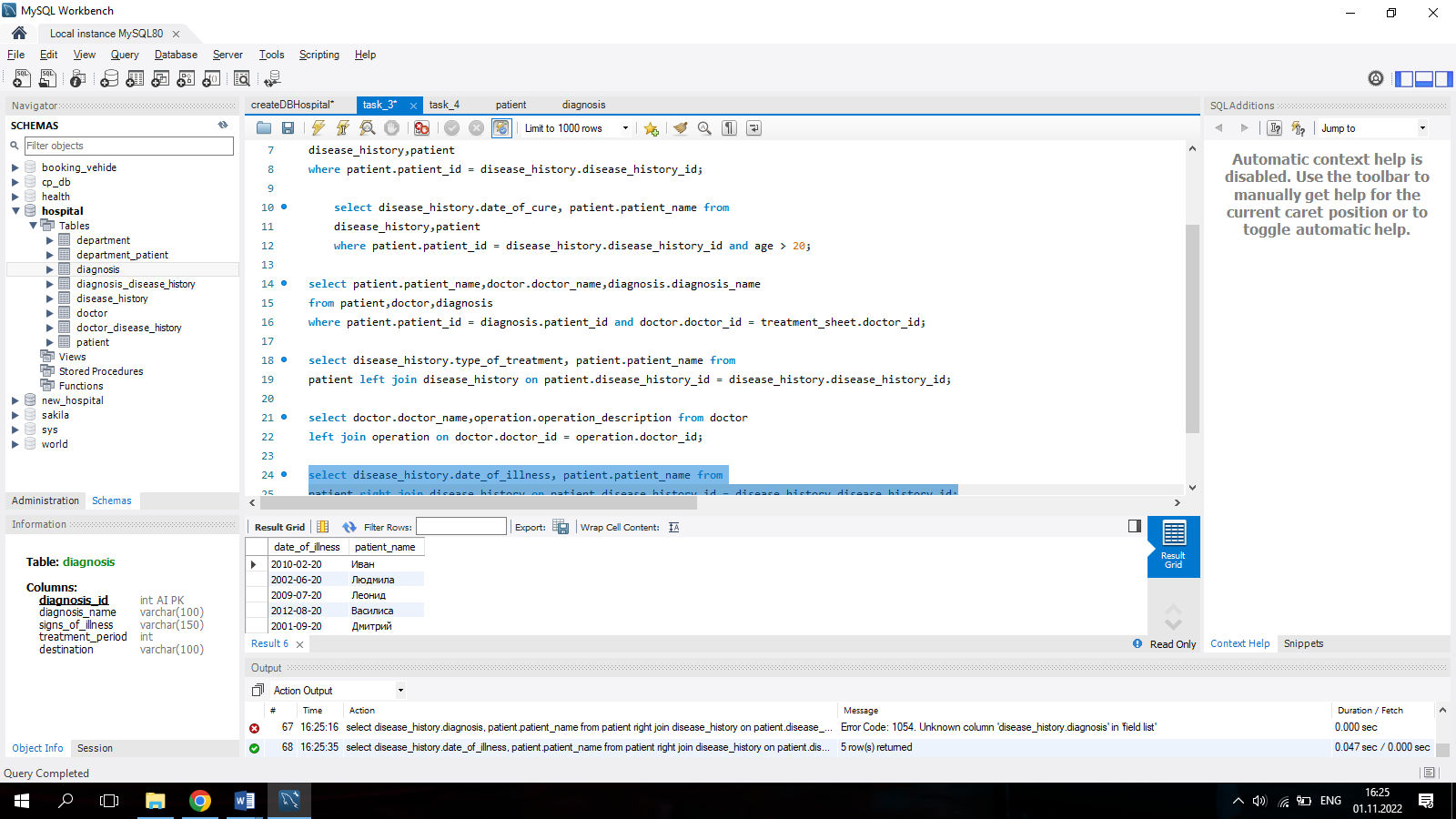
Часть 3.



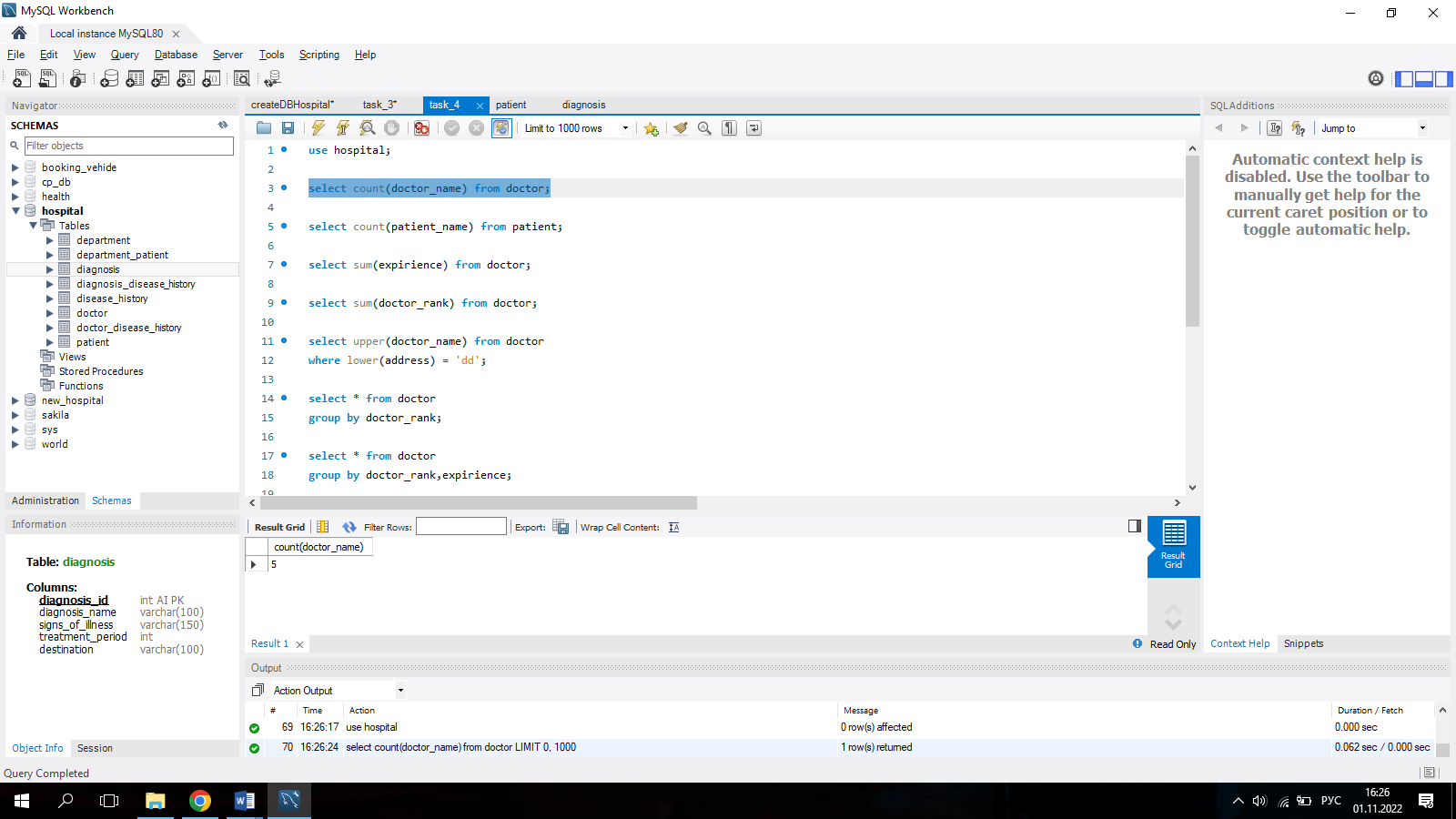


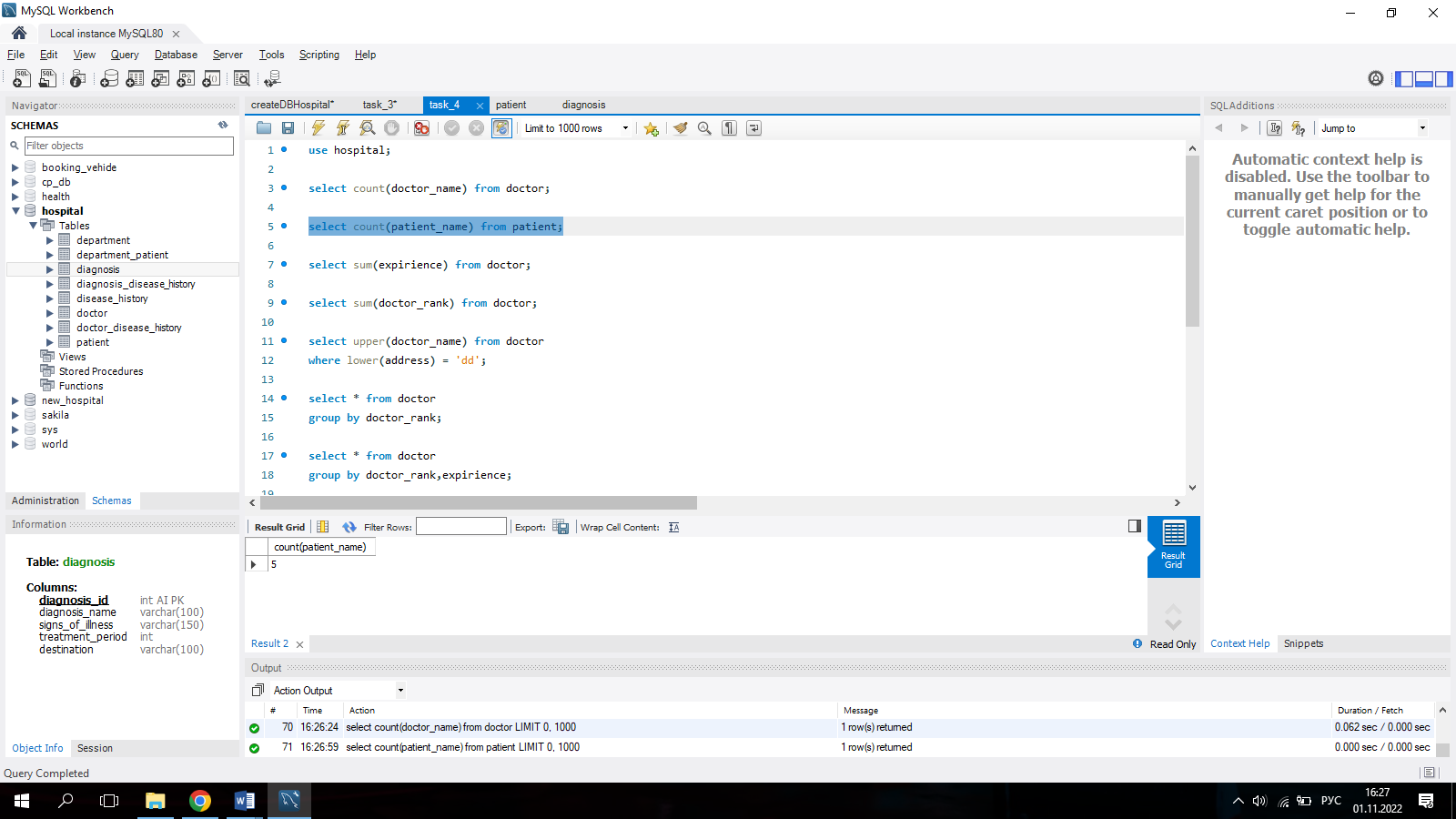


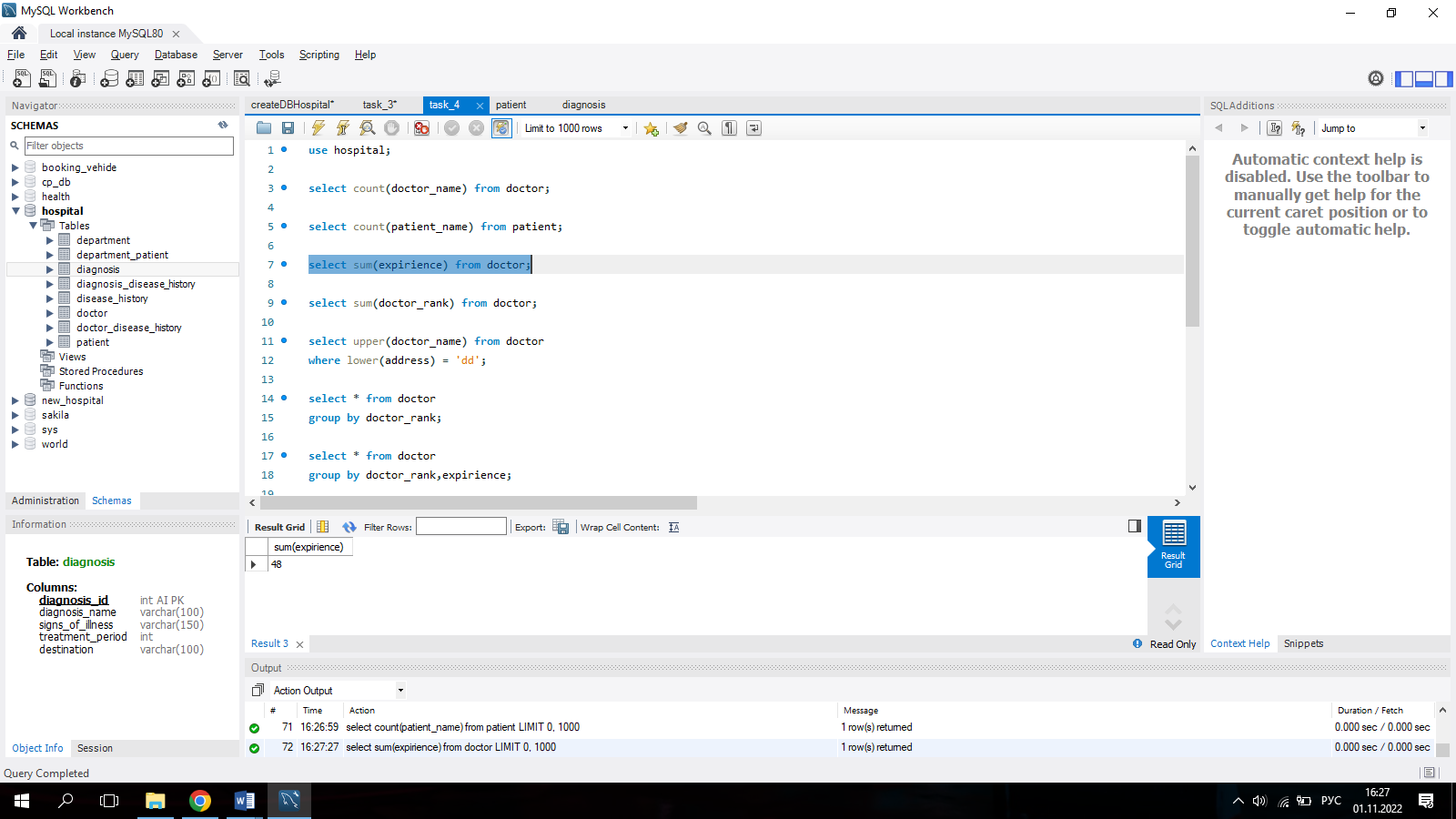


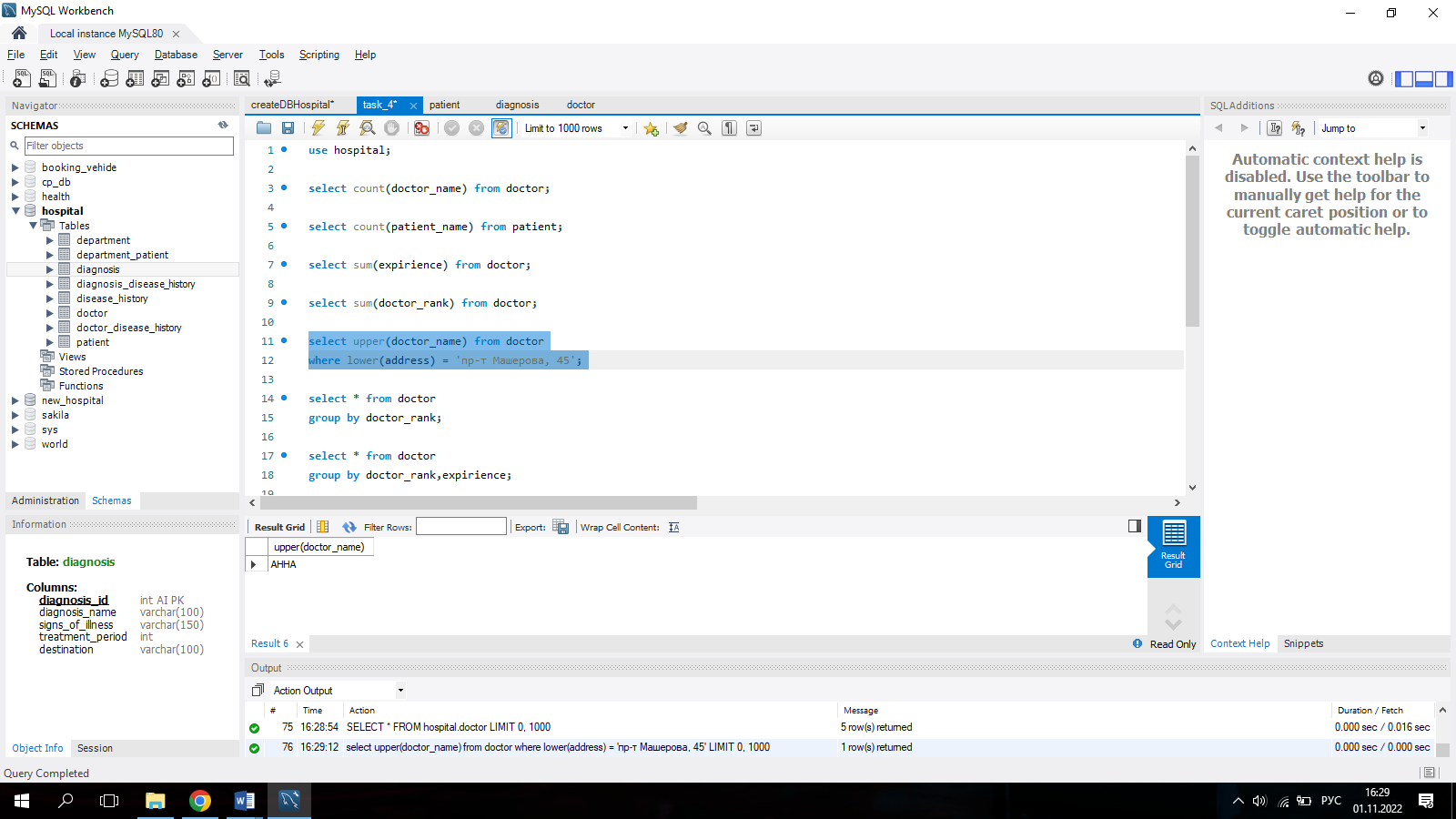


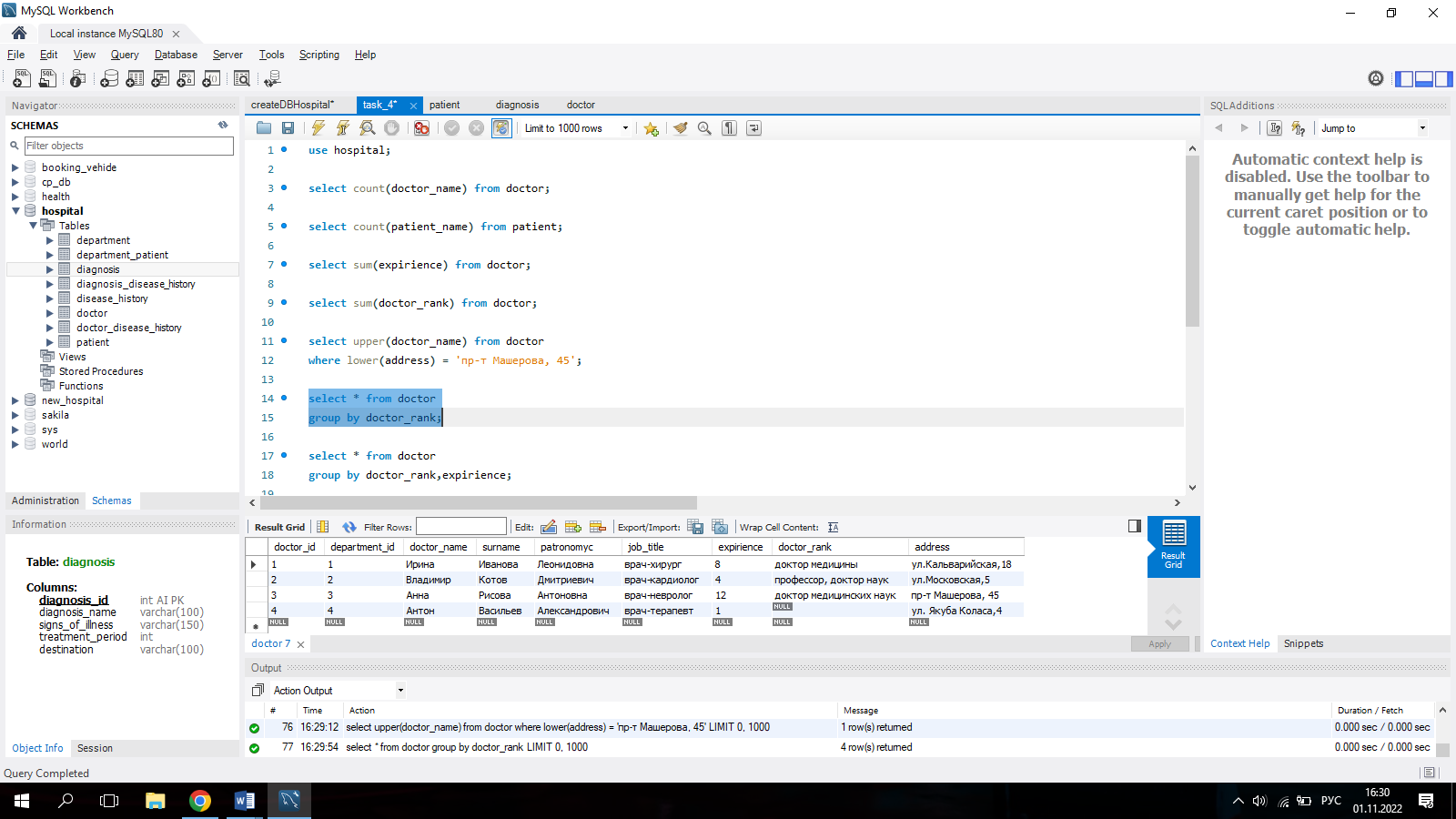
Часть 4.

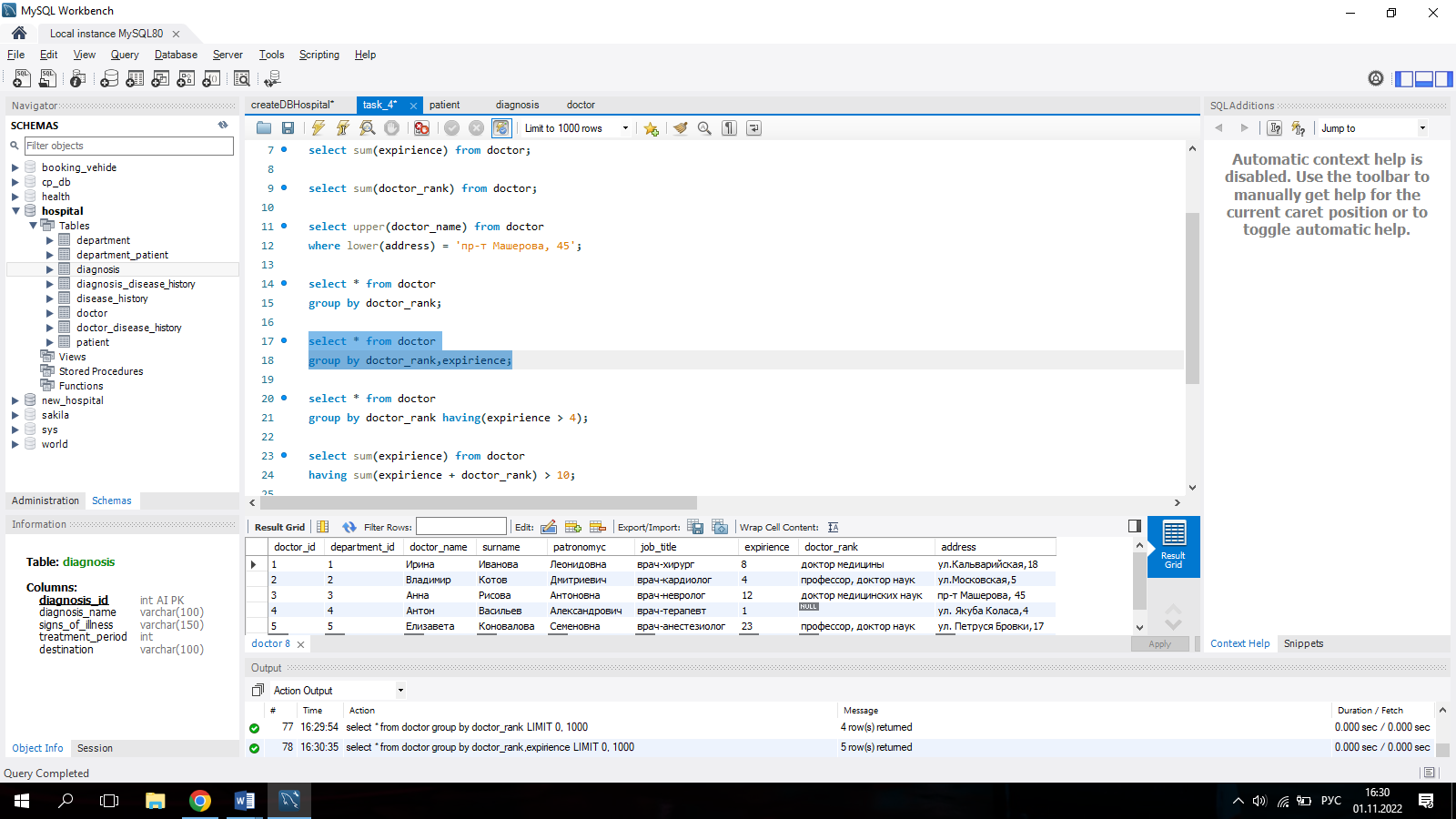


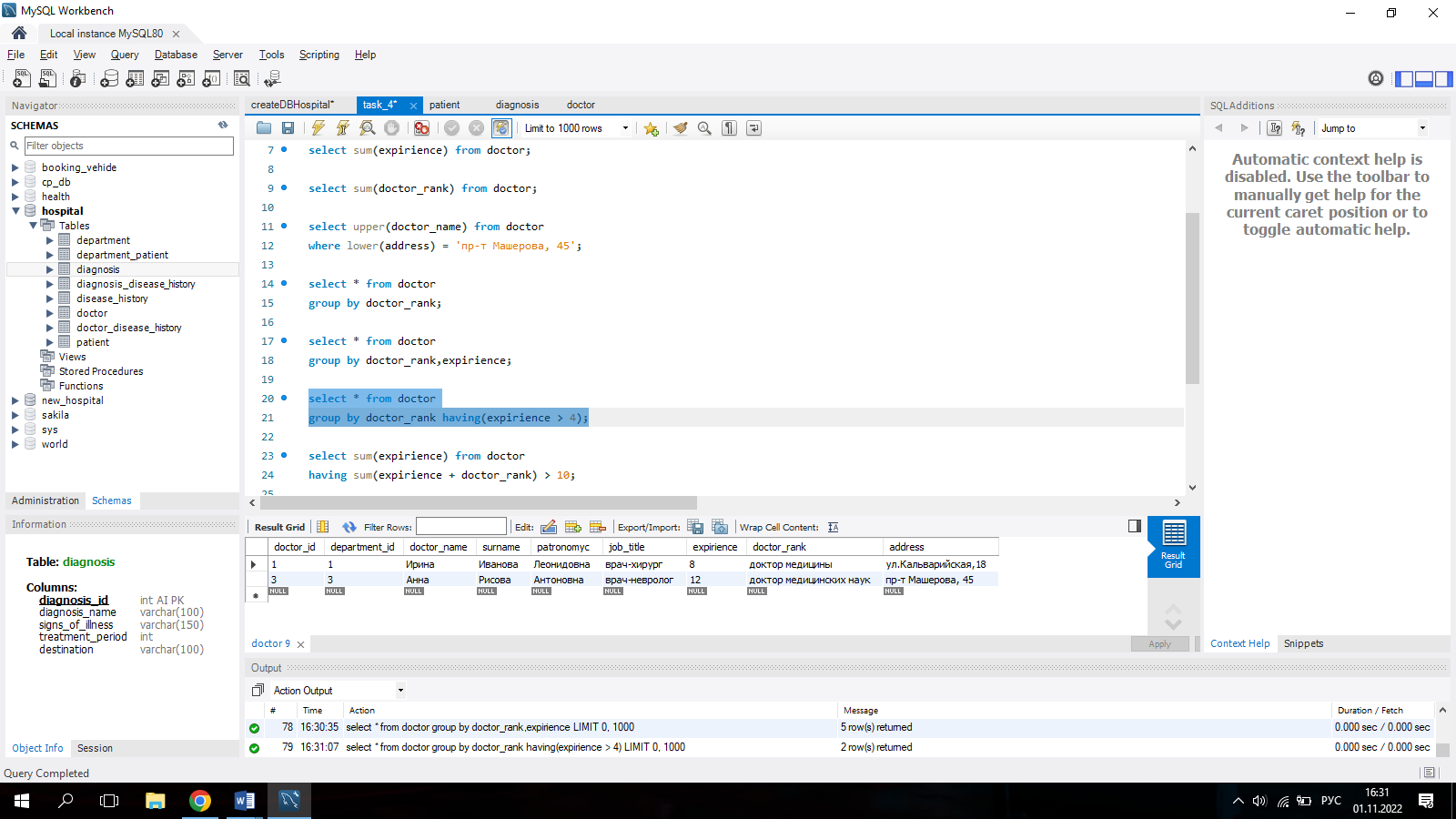


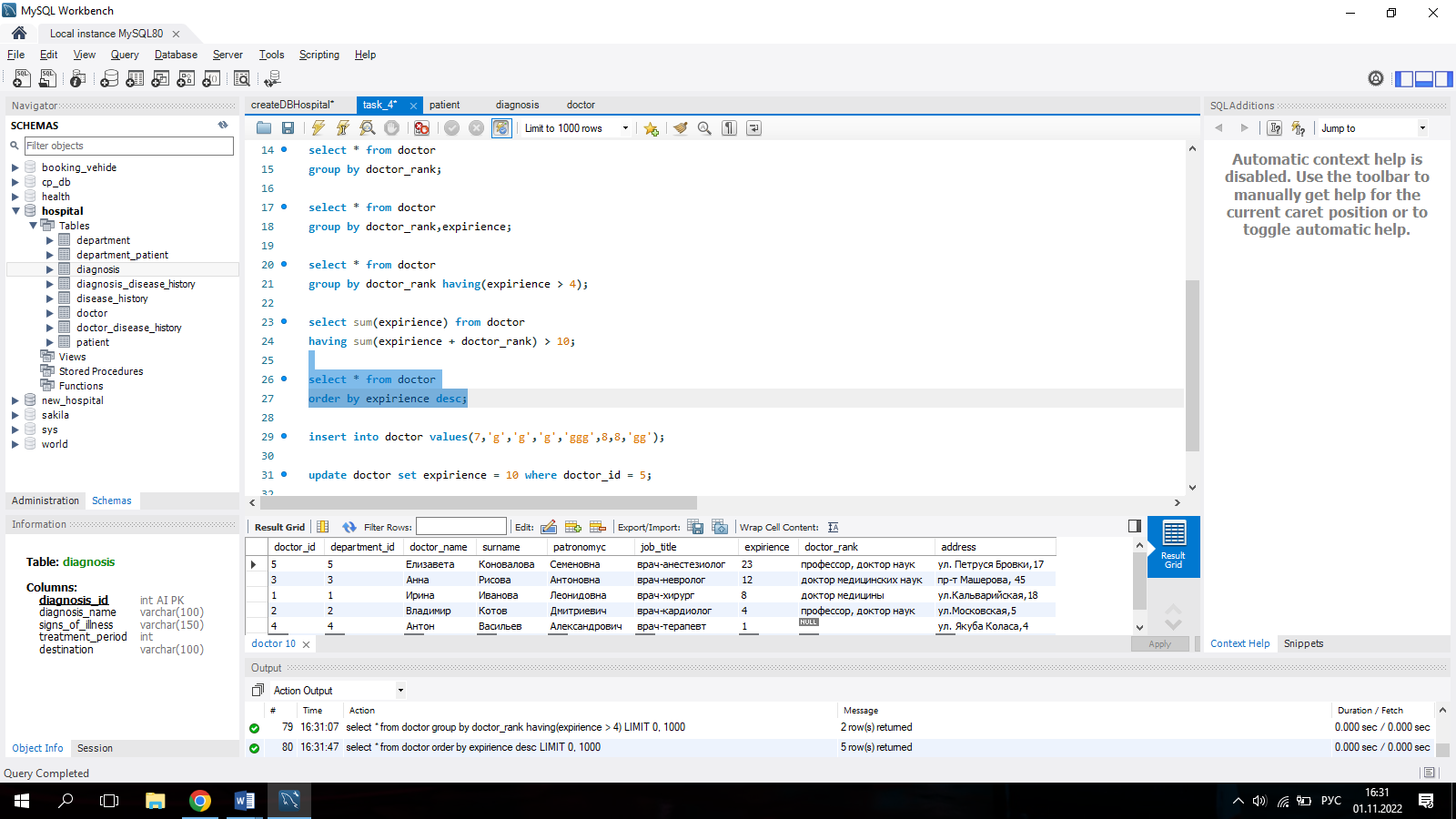




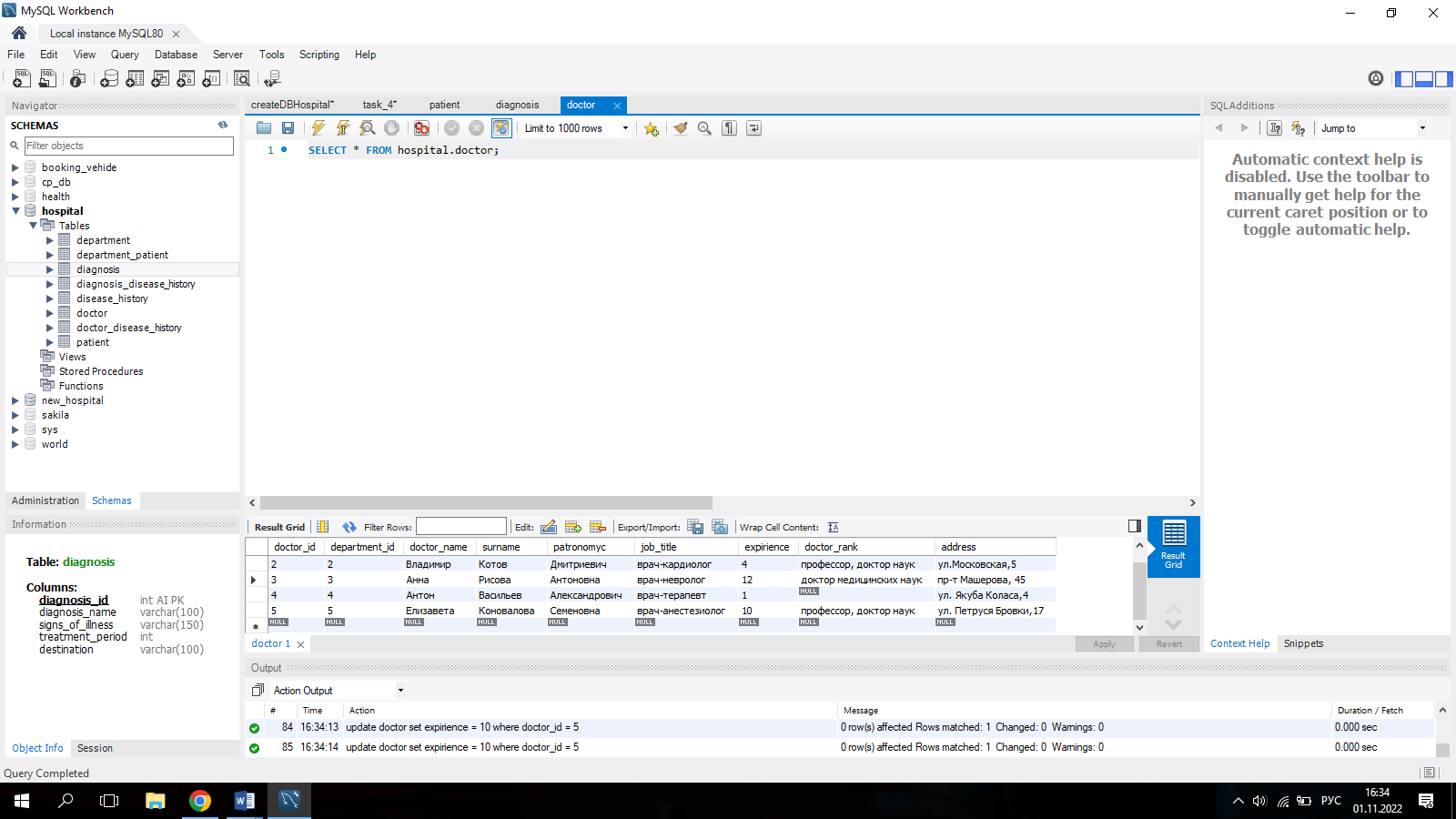








update doctor set expirience = 10 where doctor\_id = 5;



1. **Выводы**

В результате выполнения лабораторной работы все цели и задачи были выполнены. Были успешно изучены запросы Transact –SQl., изучены SQL операторы для работы с таблицами и индексами. Изучены sql- команды для создания, изменения и удаления таблиц. Изучены используемые в SQL Server типы ограничений. Изучены SQL-операторы для работы с ограничениями.