Oracle SQL Capstone Project Exam (17/07/2024)

* Code is written in MySQL format due to technical issues in Oracle XE 21c Edition. So, I had provided this code. Below the output of this code, I had provided the equivalent code in Oracle SQL format. Please consider the code given below.

MySQL Format: -

create database MHMS;

use MHMS;

-- 1. Database Schema (DDL with Constraints)

-- Create Tables patients, departments, staff, treatments

create table patients(patientid int primary key,

name varchar (100), dob date, email varchar (100) unique,

phone varchar (20));

create table departments(

departmentid int primary key,

departmentname varchar(100));

create table staff(staffid int primary key,

name varchar (100), role varchar (50), departmentid int,

foreign key (departmentid) references departments

(departmentid));

create table treatments(

treatmentid int primary key,

patientid int, staffid int,

treatmentdate date, cost decimal(10, 2),

foreign key (patientid) references patients(patientid),

foreign key (staffid) references staff(staffid),

check (cost>=0));

-- 2. Data Manipulation (DML)

-- Insert values according to given parameters in particular table

insert into patients values (1, 'john doe', '1985-05-15', 'john.doe@example.com', '555-0101');

insert into patients values (2, 'jane smith', '1990-06-20', 'jane.smith@example.com', '555-0102');

insert into patients values (3, 'alice johnson', '1975-09-25', 'alice.j@example.com', '555-0103');

insert into patients values (4, 'bob brown', '1982-01-30', 'bob.brown@example.com', '555-0104');

insert into patients values (5, 'eve white', '1995-12-12', 'eve.white@example.com', '555-0105');

select \* from patients;

insert into departments values (1, 'cardiology');

insert into departments values (2, 'neurology');

insert into departments values (3, 'orthopedics');

select \* from departments;

insert into staff values (1, 'dr. smith', 'cardiologist', 1);

insert into staff values (2, 'dr. jones', 'neurologist', 2);

insert into staff values (3, 'dr. brown', 'orthopedic surgeon', 3);

insert into staff values (4, 'nurse kelly', 'nurse', 1);

insert into staff values (5, 'nurse taylor', 'nurse', 2);

select \* from staff;

insert into treatments values (1, 1, 1, '2023-01-10', 500);

insert into treatments values (2, 2, 2, '2023-02-15', 1500);

insert into treatments values (3, 3, 3, '2023-03-20', 750);

insert into treatments values (4, 4, 1, '2023-04-05', 200);

insert into treatments values (5, 5, 2, '2023-05-30', 1200);

insert into treatments values (6, 1, 3, '2023-06-10', 300);

insert into treatments values (7, 2, 1, '2023-07-01', 800);

insert into treatments values (8, 3, 2, '2023-07-15', 950);

insert into treatments values (9, 4, 1, '2023-08-20', 1100);

insert into treatments values (10, 5, 3, '2023-09-25', 650);

select \* from treatments;

-- Update the cost of a specific treatment.

update treatments set cost=600 where treatmentid=6;

select \* from treatments;

-- Delete a patient and observe the cascading effect on associated treatments.

delete from patients where patientid=4;

-- Cannot delete or update a parent row due to failing of foreign key constraint

-- drop the existing foreign key constraint

alter table treatments drop foreign key treatments\_ibfk\_1;

-- recreate the foreign key constraint with ON DELETE CASCADE

alter table treatments

add constraint fk\_treatments\_patients

foreign key (patientid) references patients(patientid)

on delete cascade;

-- Delete a patient and observe the cascading effect on associated treatments.

delete from patients where patientid=4;

select \* from patients;

/\*

Data Retrieval using different operations

Write a query to list all patients, their associated treatments, and the staff who provided those treatments.

\*/

select p.name as patient\_name, t.treatmentdate, s.name as staff\_name

from patients p

join treatments t on p.patientid = t.patientid

join staff s on t.staffid = s.staffid;

-- Create a query to display all treatments, including patient names, staff names, and department names.

select p.name as patient\_name, s.name as staff\_name, d.departmentname, t.treatmentdate, t.cost

from treatments t

join patients p on t.patientid = p.patientid

join staff s on t.staffid = s.staffid

join departments d on s.departmentid = d.departmentid;

/\*

4. Subqueries and Correlated Subqueries

Use a subquery to find patients who have received treatments from a specific department (provided DepartmentID).

\*/

select distinct p.\*

from patients p

join treatments t on p.patientid = t.patientid

join staff s on t.staffid = s.staffid

where s.departmentid = 1;

-- Use a correlated subquery to find staff who have treated more than a specified number of patients (e.g., 3).

select \*

from staff s

where (select count(\*) from treatments t

where t.staffid = s.staffid) > 2;

/\*

5. Aggregate Functions with GROUP BY and HAVING [5 Marks]

Calculate the total cost of treatments per patient and filter out patients

with a total treatment cost of less than Rs.1000

\*/

select p.name as patient\_name, sum(t.cost) as total\_cost

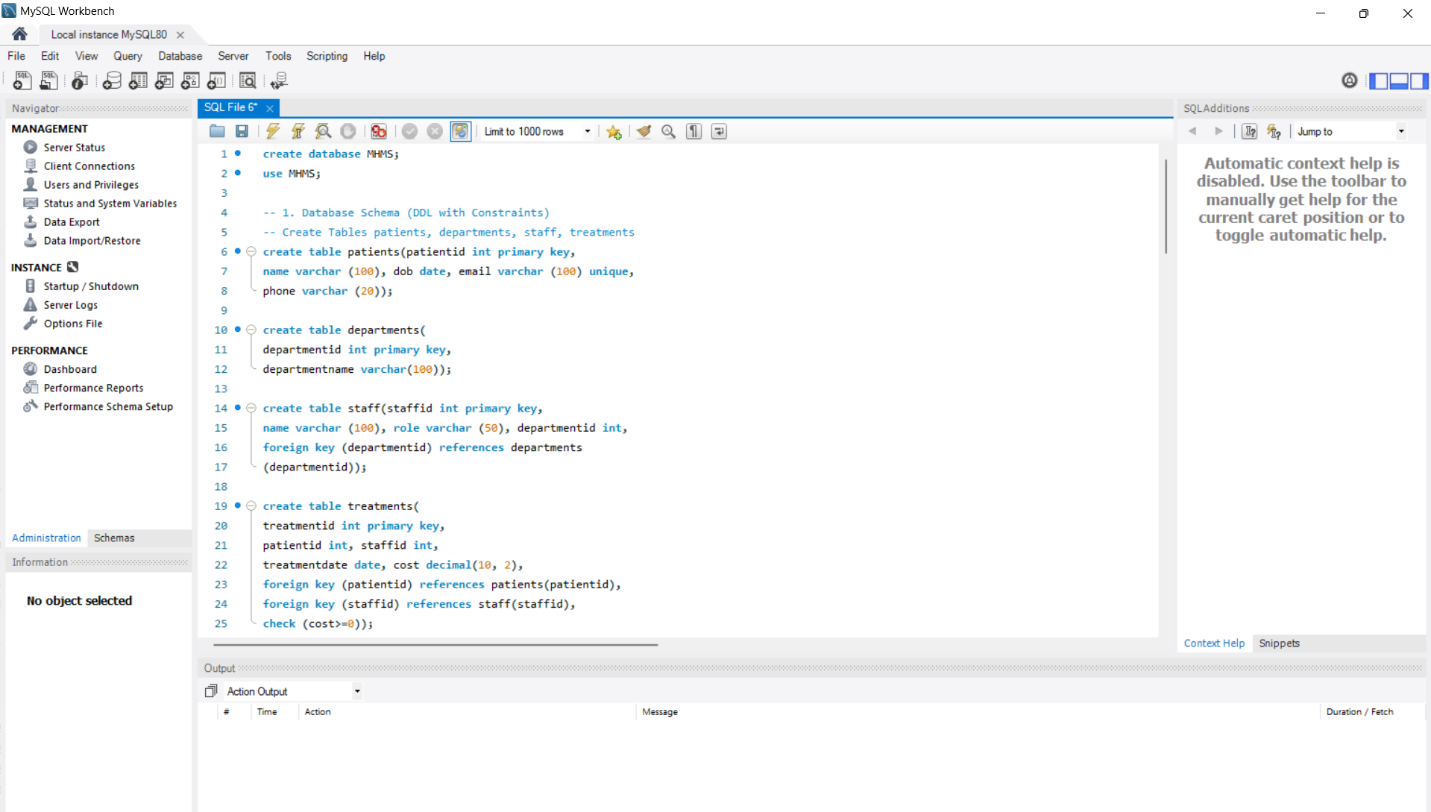
from patients p

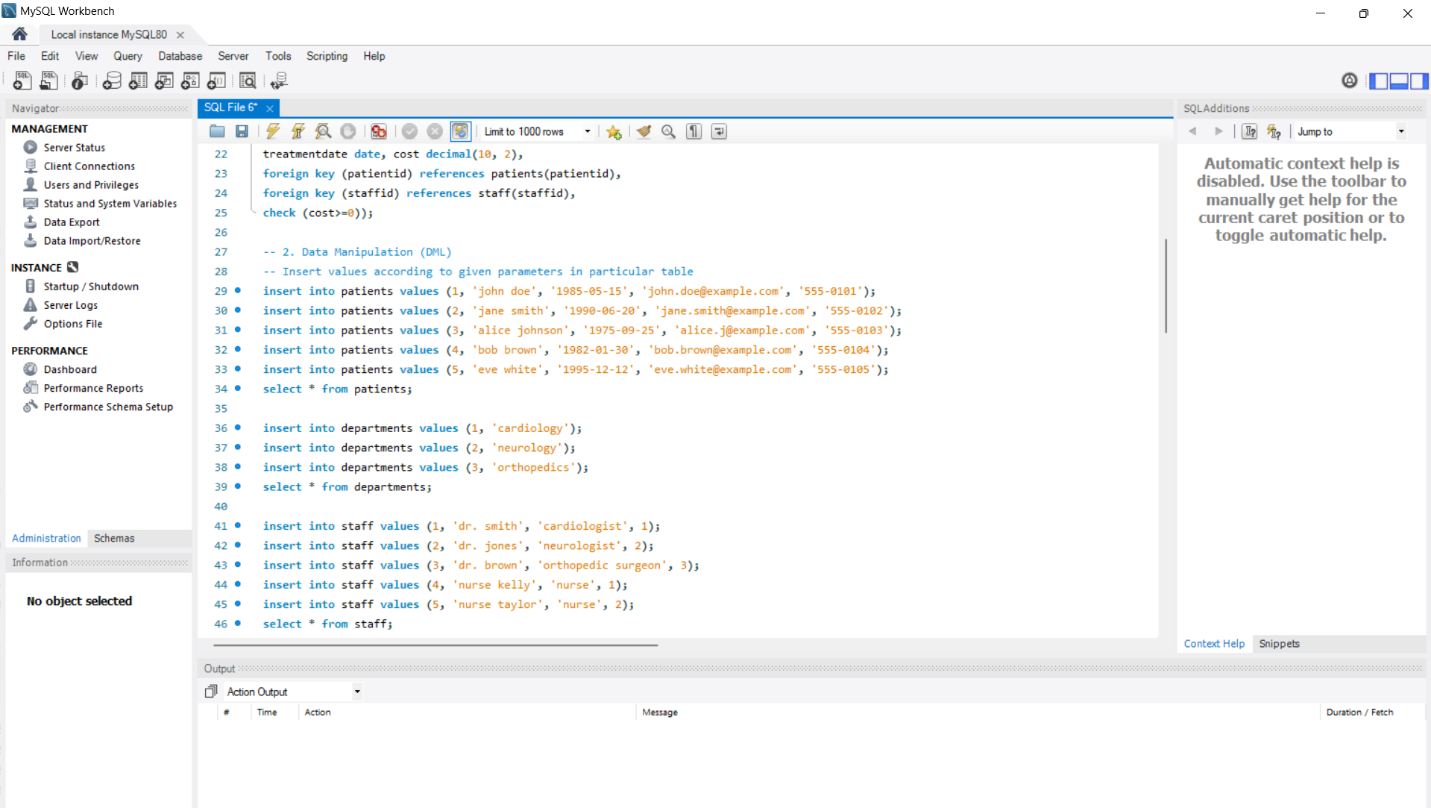
join treatments t on p.patientid = t.patientid

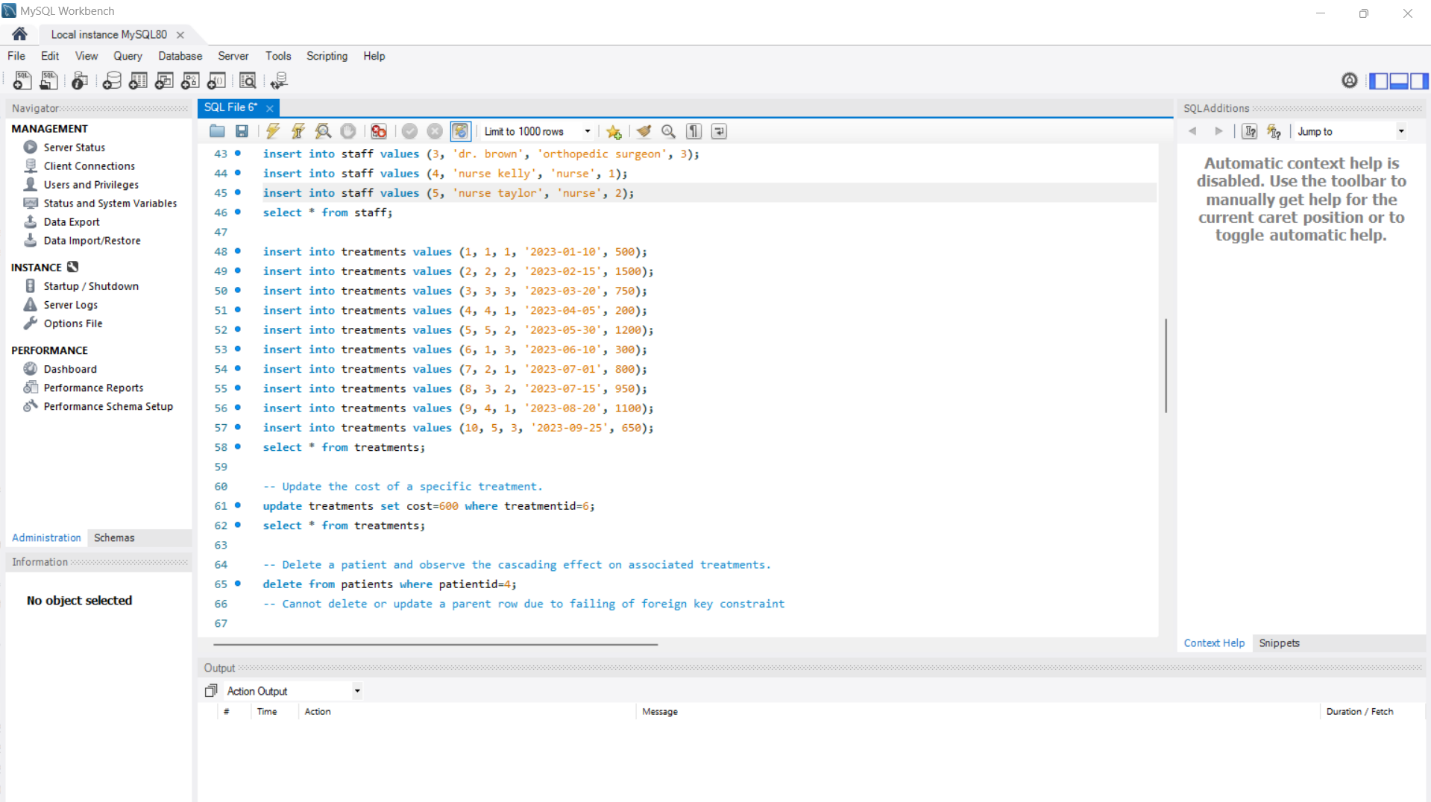
group by p.name

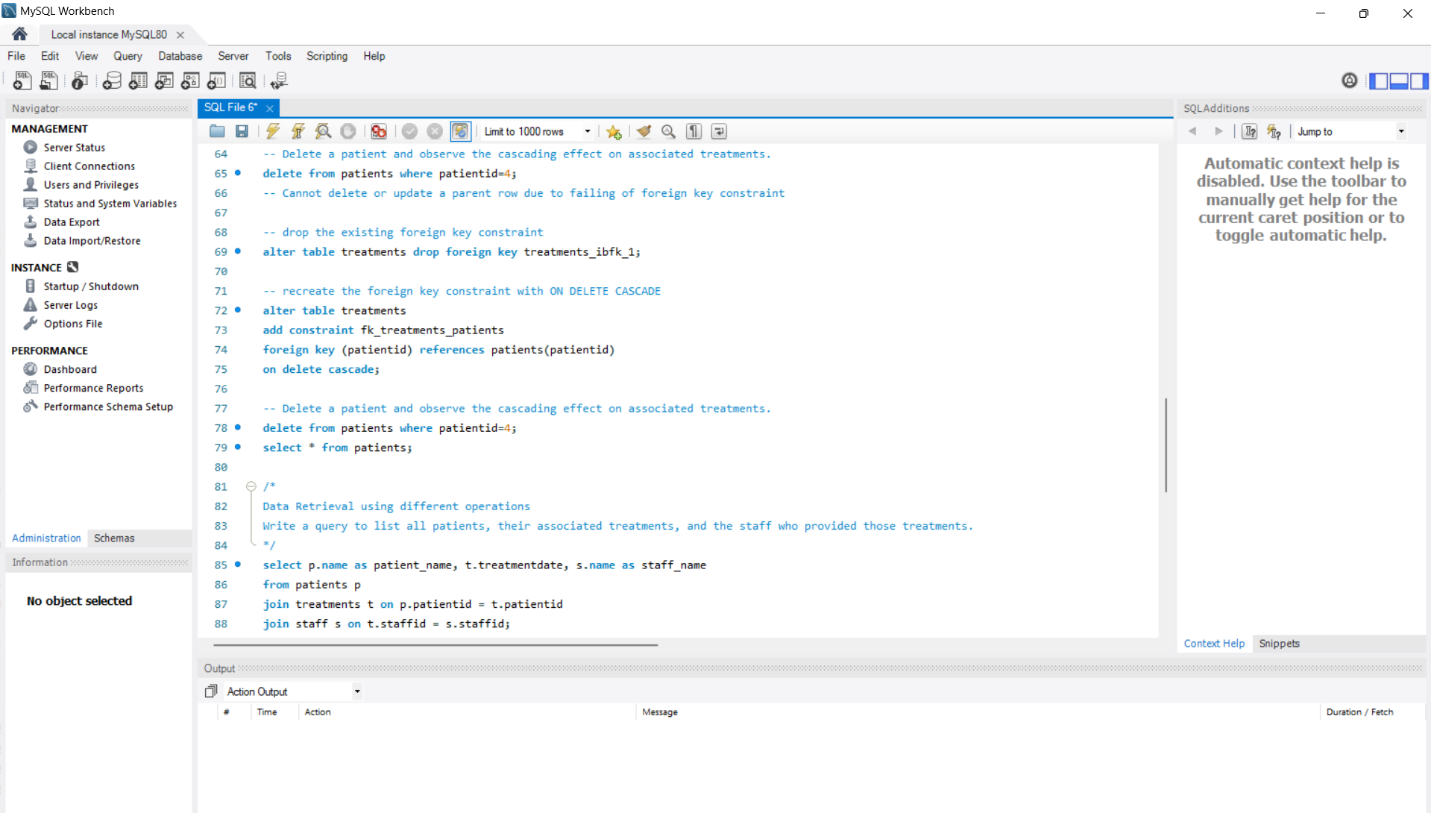
having sum(t.cost) >= 1000;

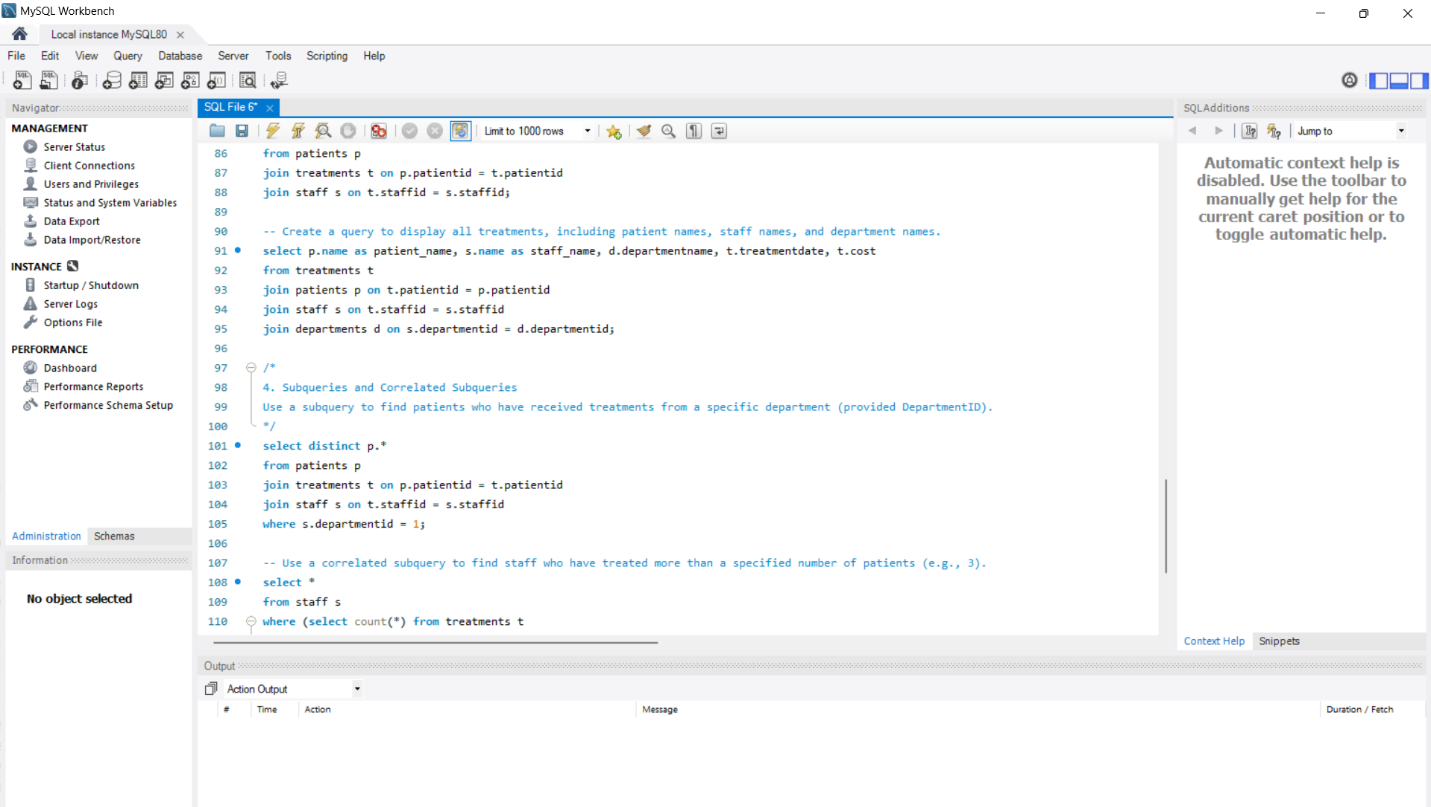
OUTPUT: -

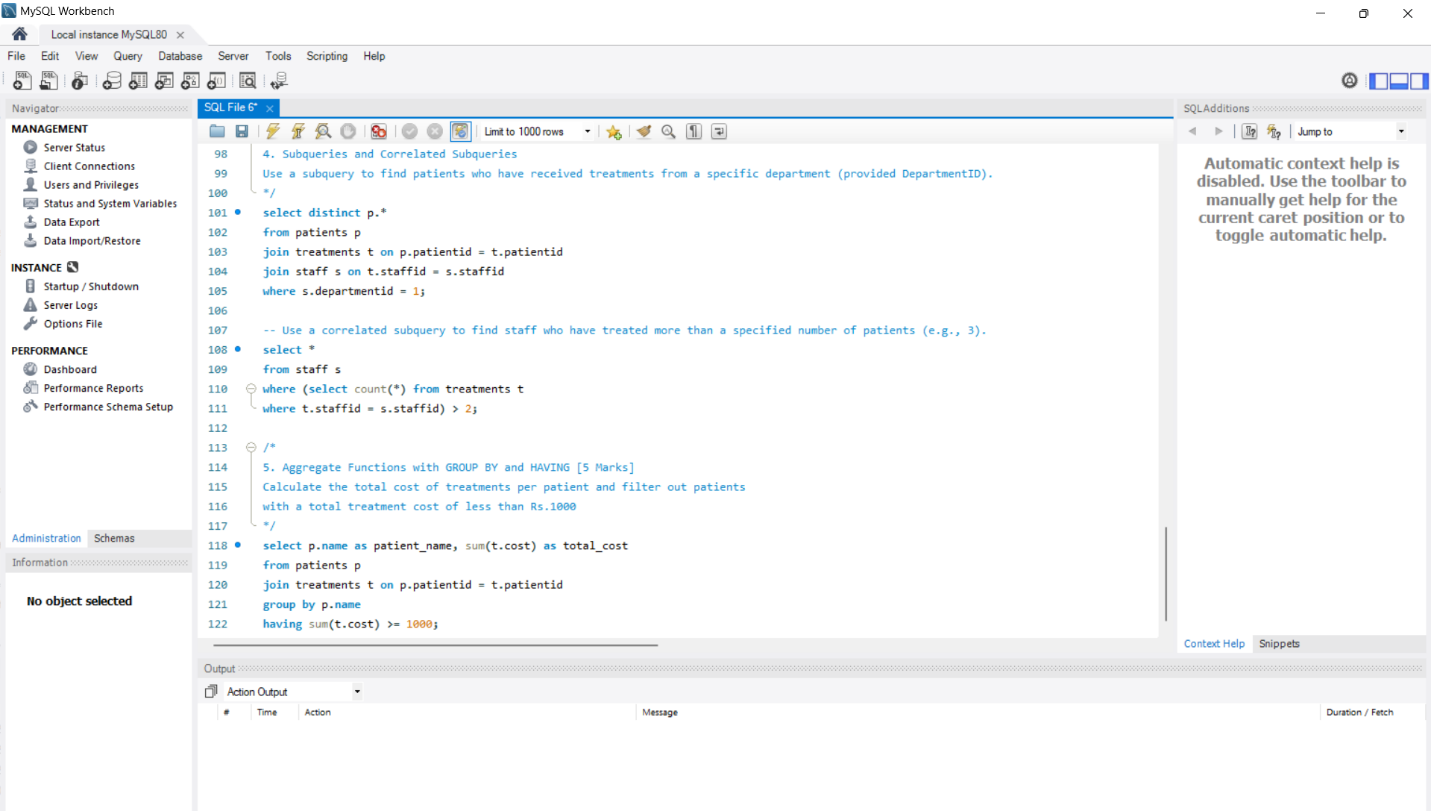


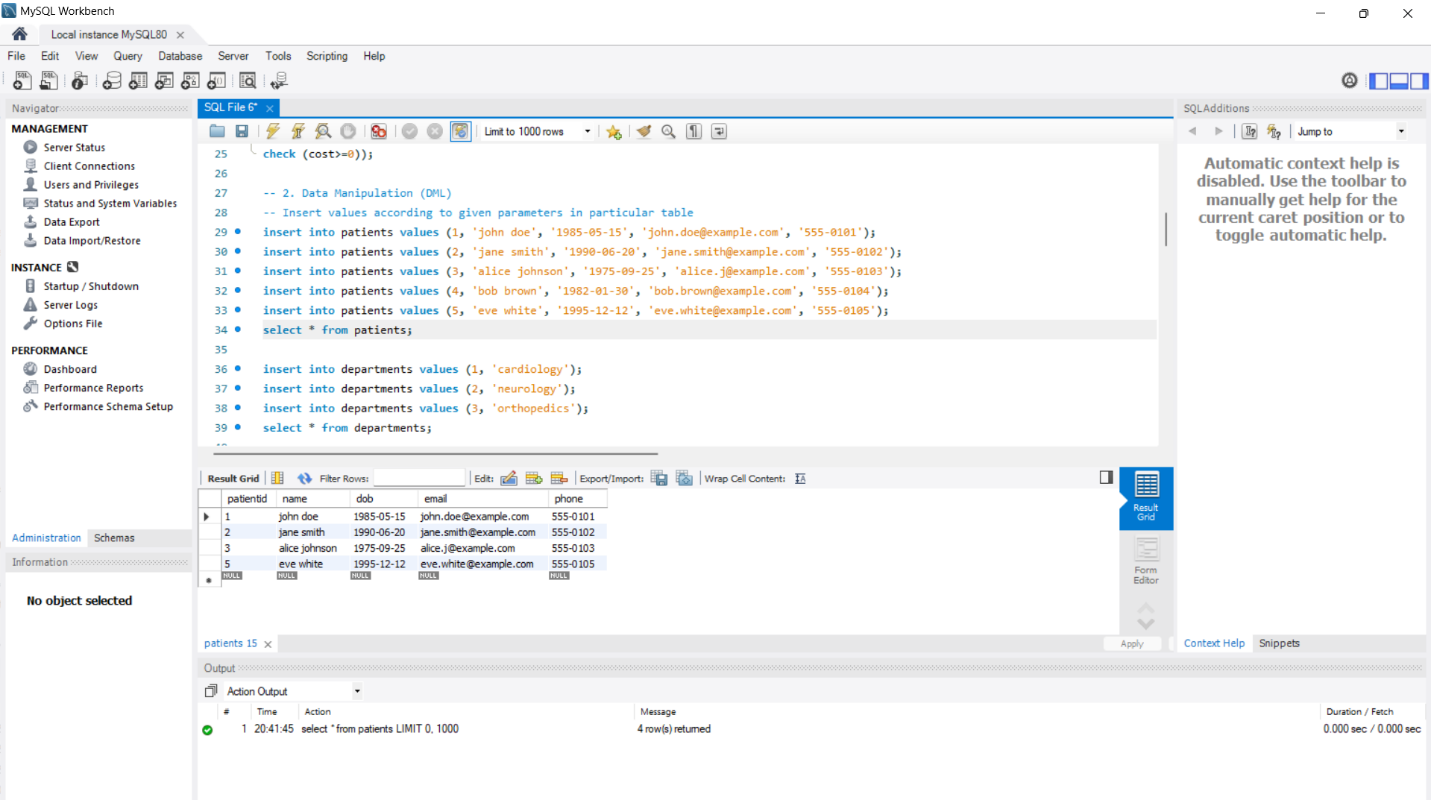


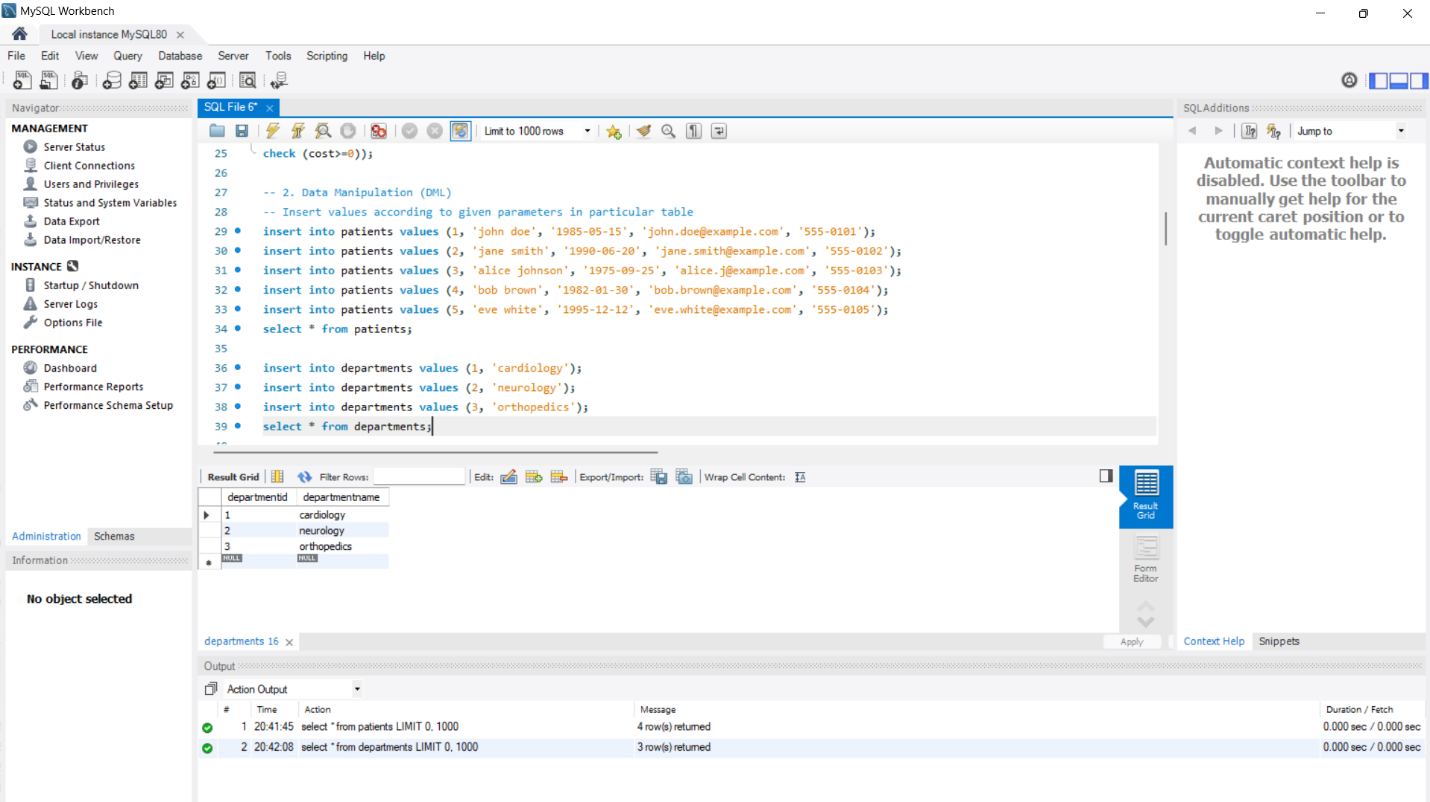




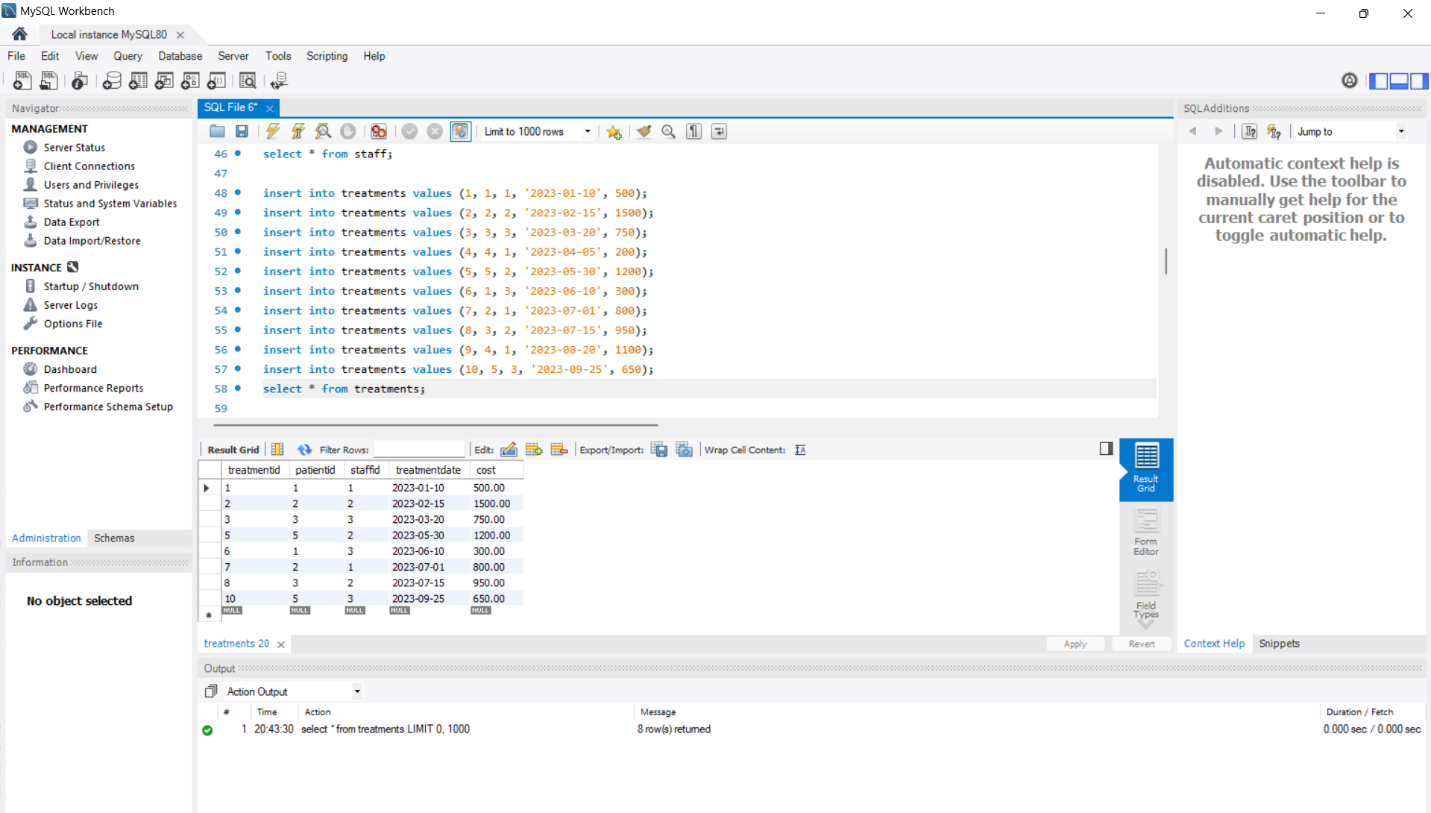


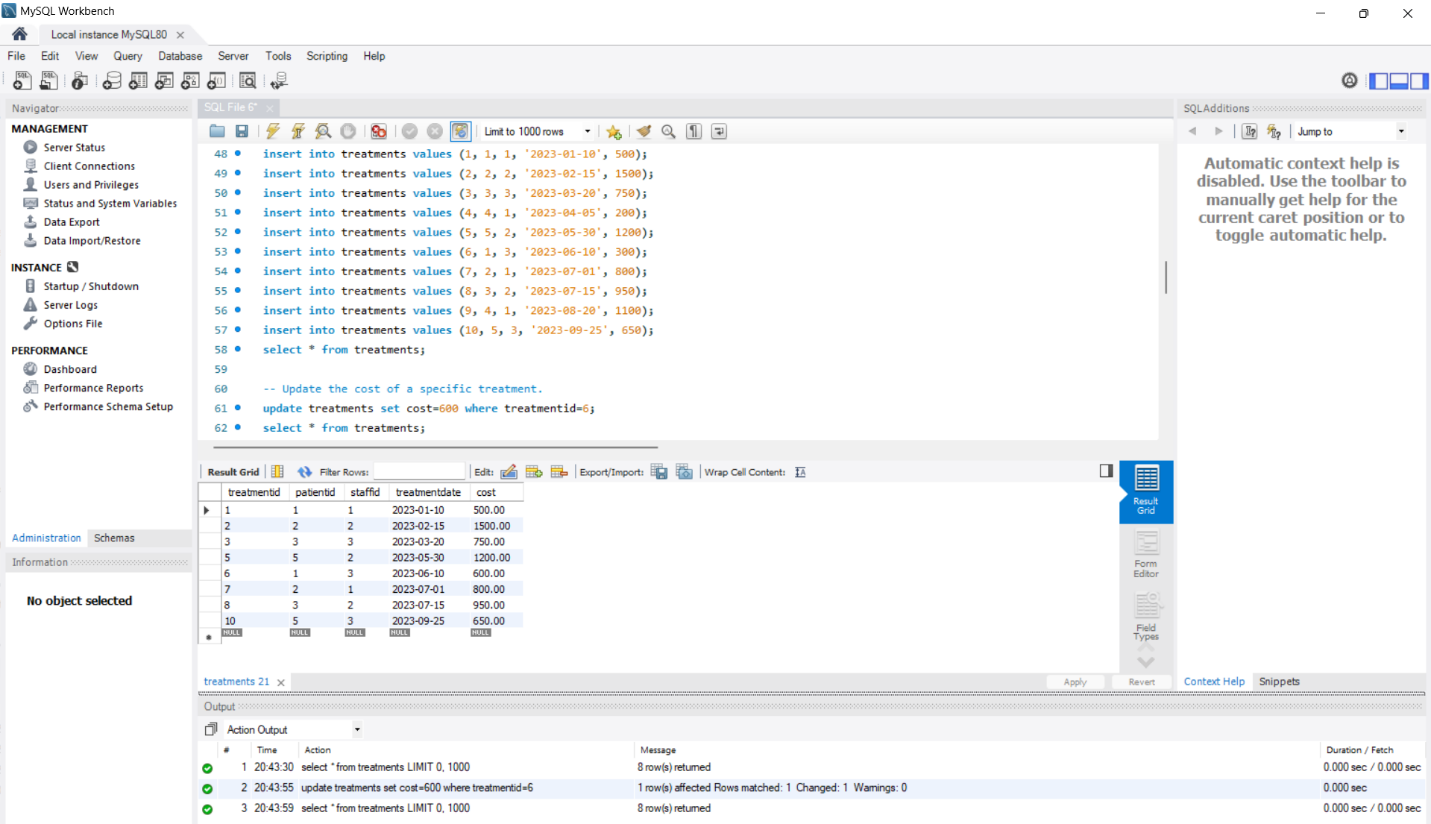


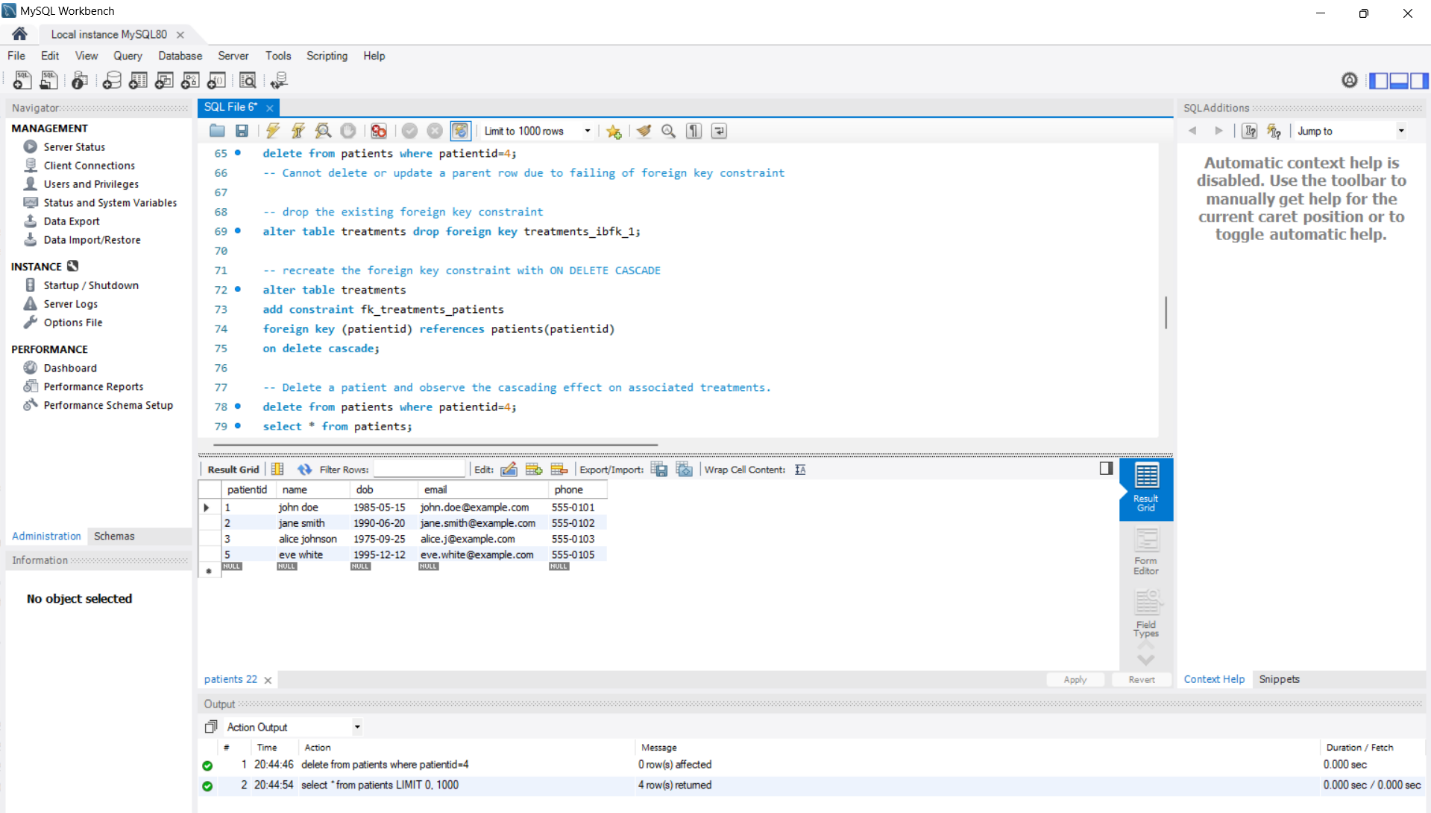


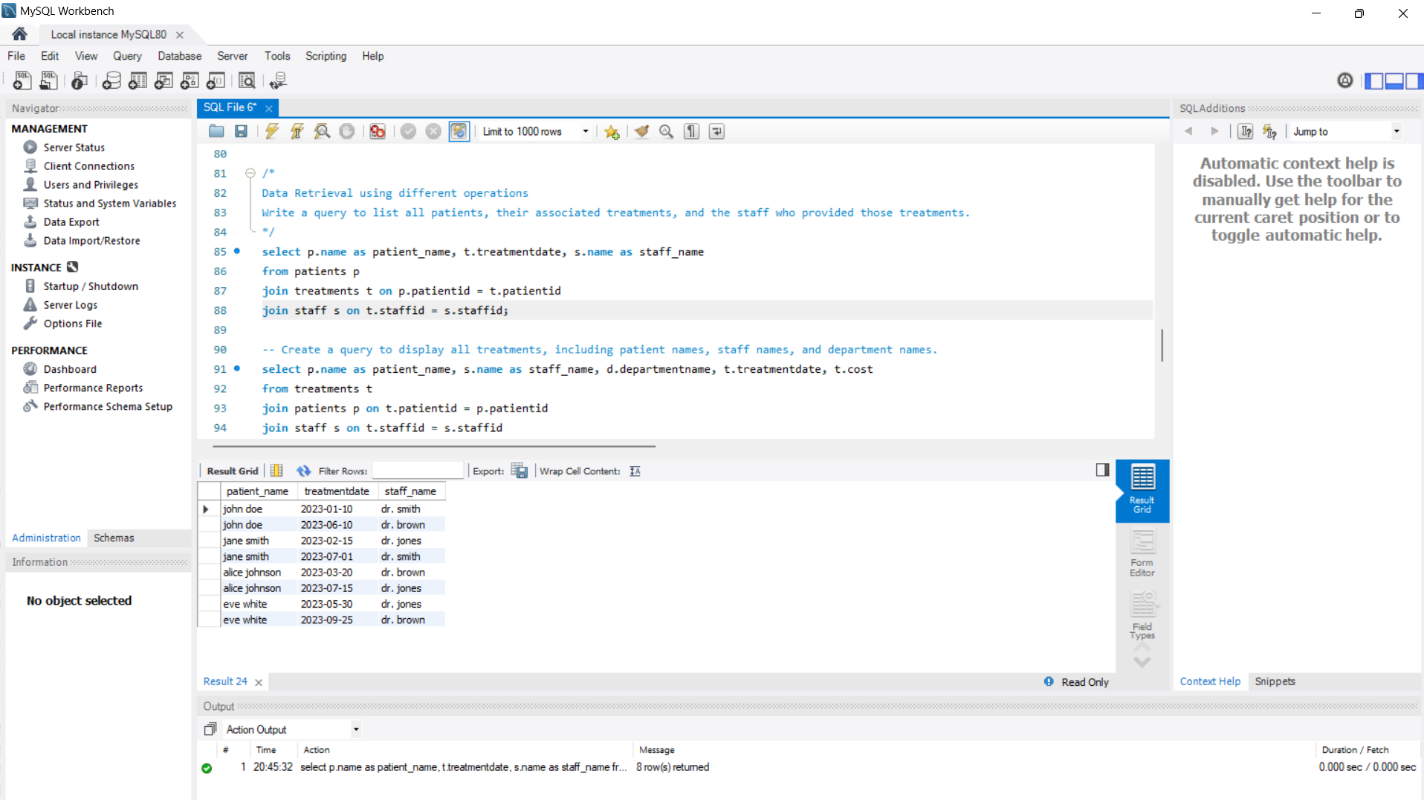


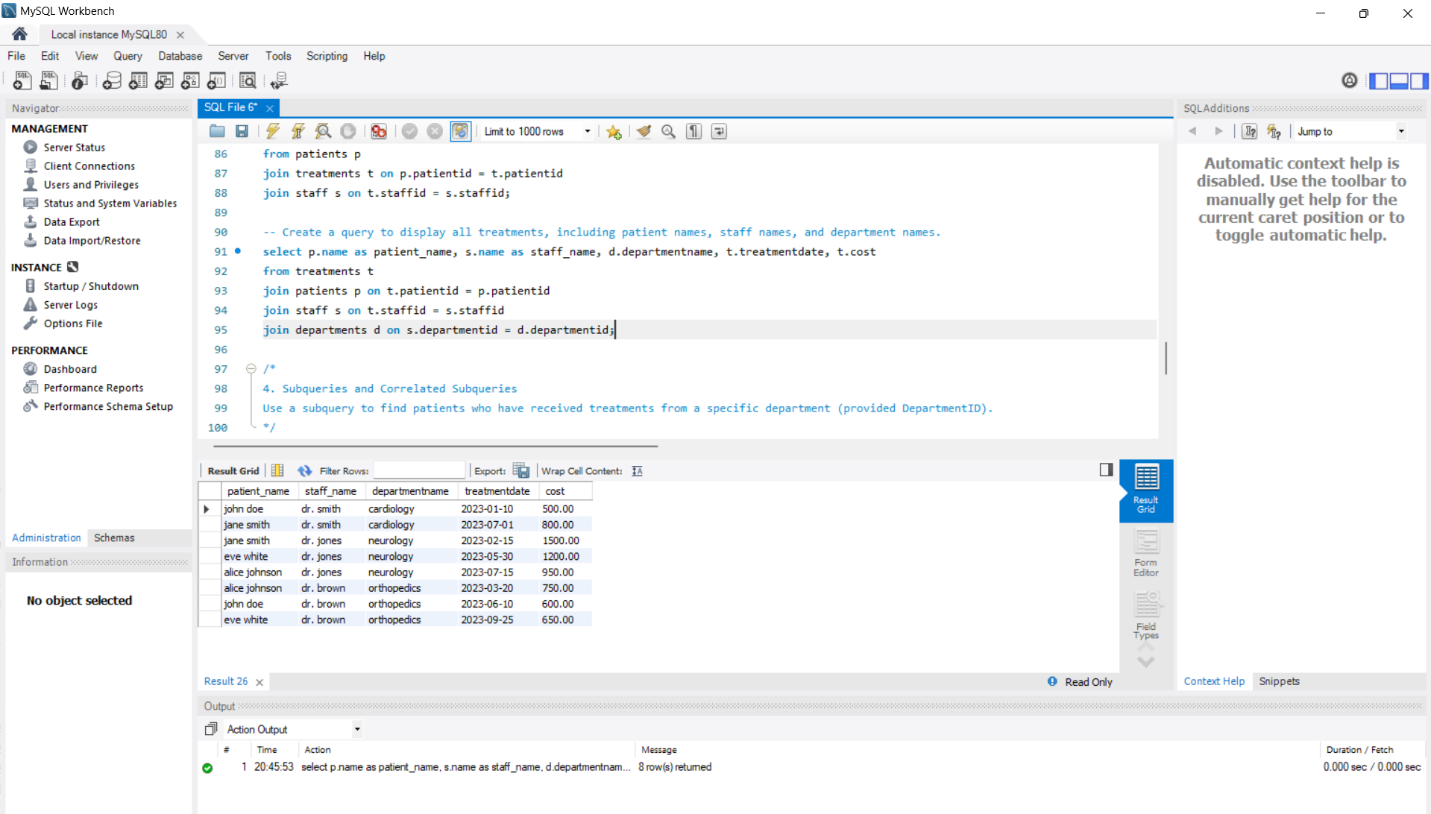


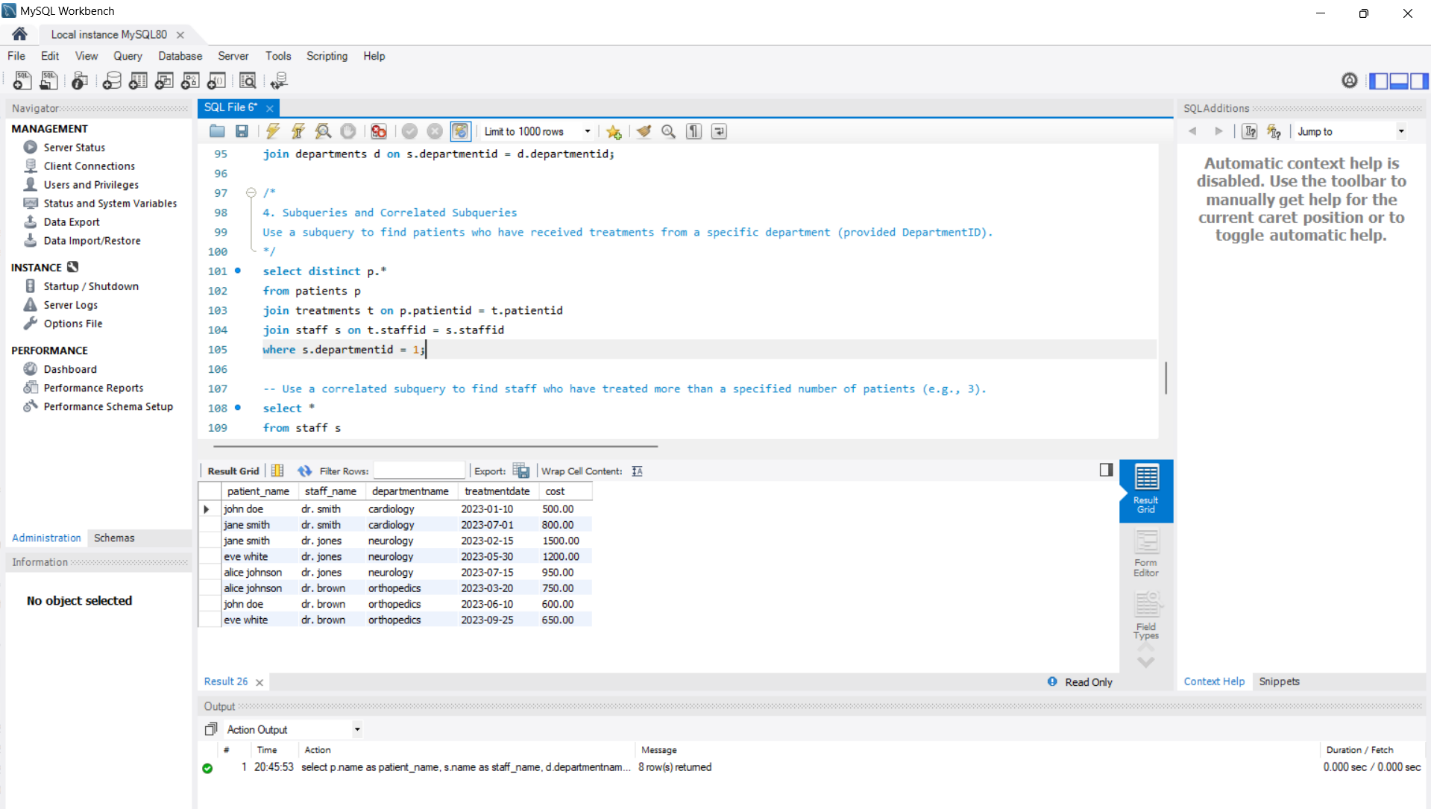


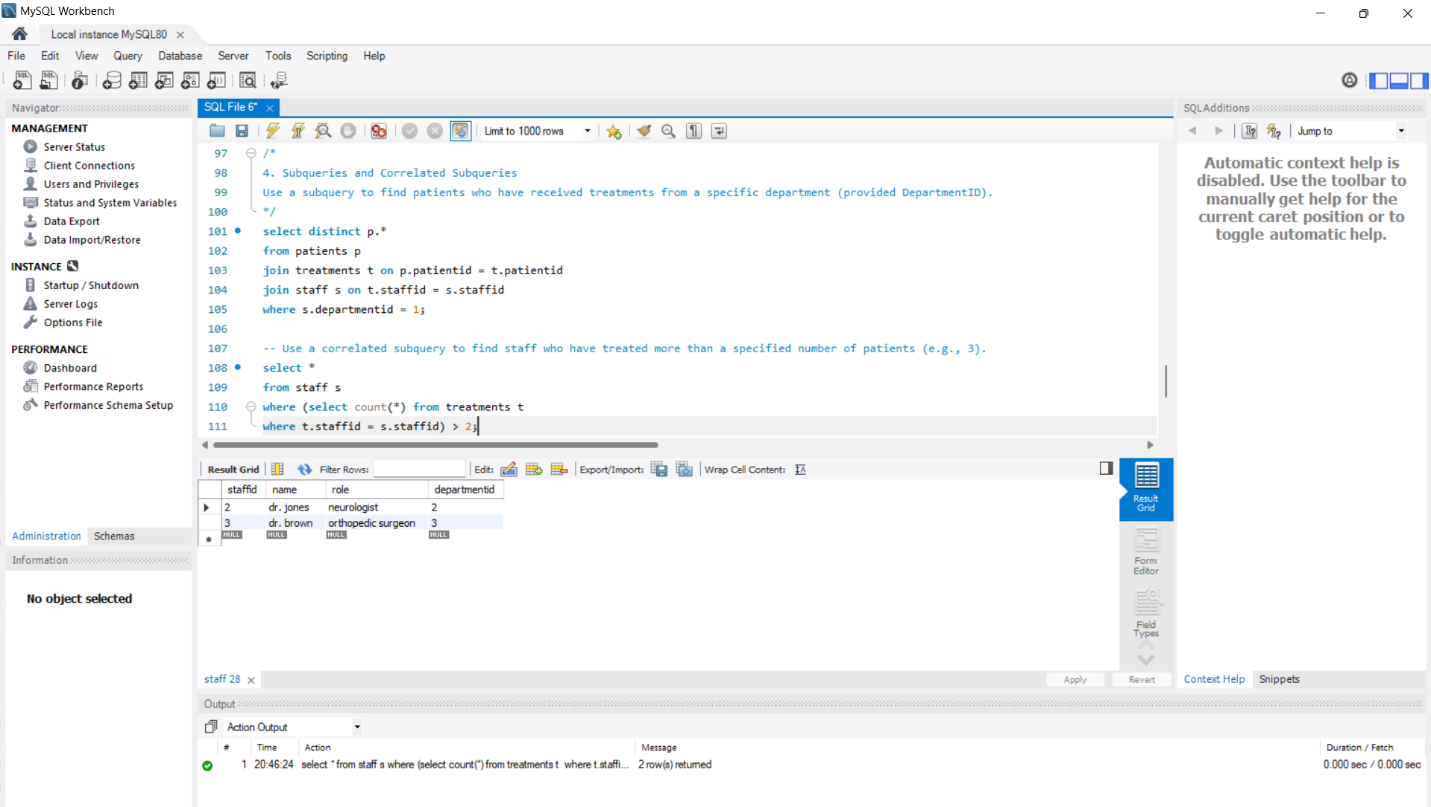


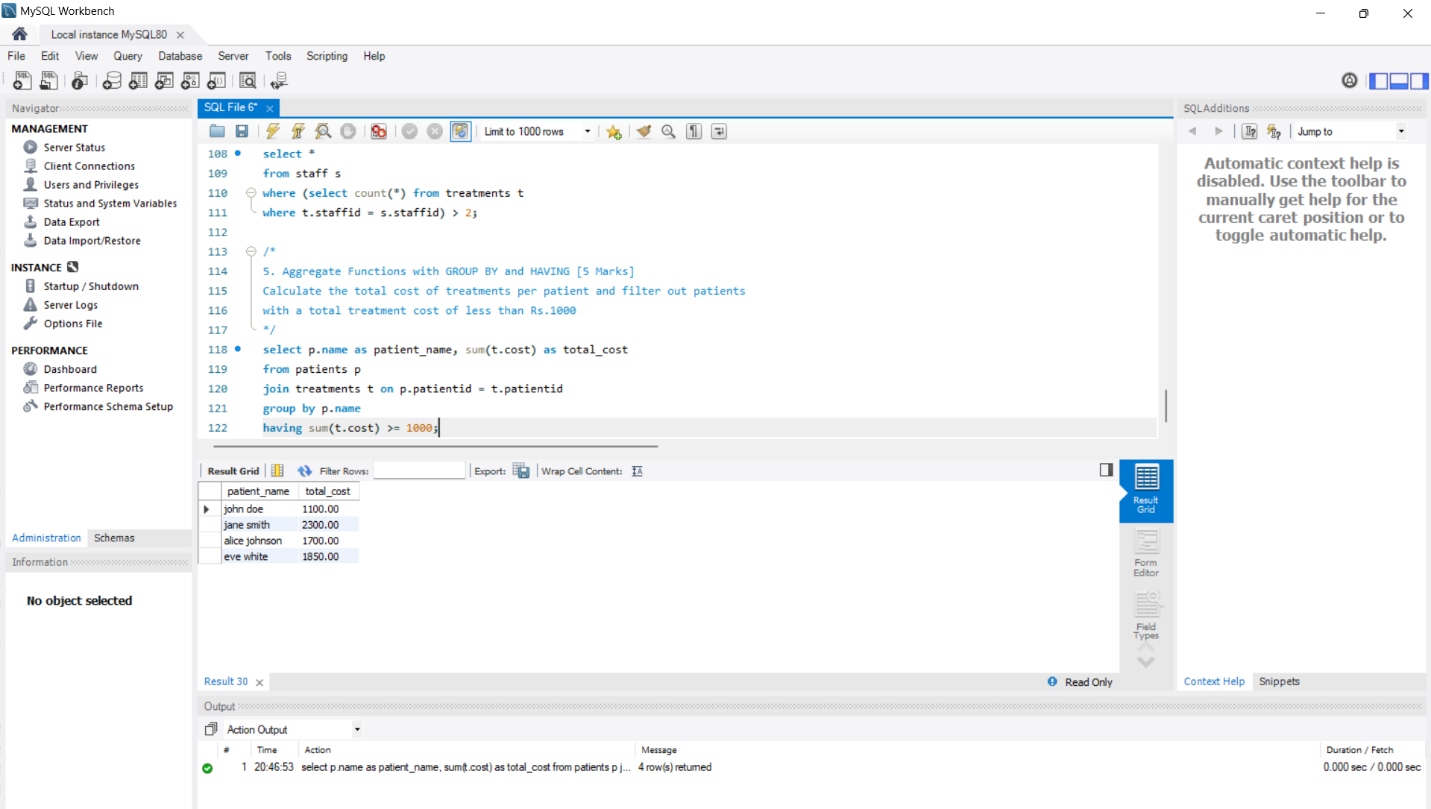












Oracle Database XE 21c Format: -

-- Create Tables: patients, departments, staff, treatments

-- Patients table

create table patients (

patientid number primary key,

name varchar2(100),

dob date,

email varchar2(100) unique,

phone varchar2(20)

);

-- Departments table

create table departments (

departmentid number primary key,

departmentname varchar2(100)

);

-- Staff table

create table staff (

staffid number primary key,

name varchar2(100),

role varchar2(50),

departmentid number,

constraint fk\_staff\_department foreign key (departmentid) references departments(departmentid)

);

-- Treatments table

create table treatments (

treatmentid number primary key,

patientid number,

staffid number,

treatmentdate date,

cost number(10, 2),

constraint fk\_treatments\_patients foreign key (patientid) references patients(patientid) on delete cascade,

constraint fk\_treatments\_staff foreign key (staffid) references staff(staffid),

constraint chk\_cost\_nonnegative check (cost >= 0)

);

-- Insert sample data into patients table

insert into patients values (1, 'john doe', to\_date('1985-05-15', 'yyyy-mm-dd'), 'john.doe@example.com', '555-0101');

insert into patients values (2, 'jane smith', to\_date('1990-06-20', 'yyyy-mm-dd'), 'jane.smith@example.com', '555-0102');

insert into patients values (3, 'alice johnson', to\_date('1975-09-25', 'yyyy-mm-dd'), 'alice.j@example.com', '555-0103');

insert into patients values (4, 'bob brown', to\_date('1982-01-30', 'yyyy-mm-dd'), 'bob.brown@example.com', '555-0104');

insert into patients values (5, 'eve white', to\_date('1995-12-12', 'yyyy-mm-dd'), 'eve.white@example.com', '555-0105');

-- Insert sample data into departments table

insert into departments values (1, 'cardiology');

insert into departments values (2, 'neurology');

insert into departments values (3, 'orthopedics');

-- Insert sample data into staff table

insert into staff values (1, 'dr. smith', 'cardiologist', 1);

insert into staff values (2, 'dr. jones', 'neurologist', 2);

insert into staff values (3, 'dr. brown', 'orthopedic surgeon', 3);

insert into staff values (4, 'nurse kelly', 'nurse', 1);

insert into staff values (5, 'nurse taylor', 'nurse', 2);

-- Insert sample data into treatments table

insert into treatments values (1, 1, 1, to\_date('2023-01-10', 'yyyy-mm-dd'), 500);

insert into treatments values (2, 2, 2, to\_date('2023-02-15', 'yyyy-mm-dd'), 1500);

insert into treatments values (3, 3, 3, to\_date('2023-03-20', 'yyyy-mm-dd'), 750);

insert into treatments values (4, 4, 1, to\_date('2023-04-05', 'yyyy-mm-dd'), 200);

insert into treatments values (5, 5, 2, to\_date('2023-05-30', 'yyyy-mm-dd'), 1200);

insert into treatments values (6, 1, 3, to\_date('2023-06-10', 'yyyy-mm-dd'), 300);

insert into treatments values (7, 2, 1, to\_date('2023-07-01', 'yyyy-mm-dd'), 800);

insert into treatments values (8, 3, 2, to\_date('2023-07-15', 'yyyy-mm-dd'), 950);

insert into treatments values (9, 4, 1, to\_date('2023-08-20', 'yyyy-mm-dd'), 1100);

insert into treatments values (10, 5, 3, to\_date('2023-09-25', 'yyyy-mm-dd'), 650);

-- Update the cost of a specific treatment

update treatments set cost = 600 where treatmentid = 6;

-- Delete a patient and observe the cascading effect on associated treatments

-- Note: Ensure ON DELETE CASCADE is enabled on the foreign key constraint (already defined above)

-- Delete patient with patientid = 4

delete from patients where patientid = 4;

select \* from treatments;

/\*

Data Retrieval using different operations

Write a query to list all patients, their associated treatments, and the staff who provided those treatments.

\*/

select p.name as patient\_name, t.treatmentdate, s.name as staff\_name

from patients p

join treatments t on p.patientid = t.patientid

join staff s on t.staffid = s.staffid;

-- Create a query to display all treatments, including patient names, staff names, and department names

select p.name as patient\_name, s.name as staff\_name, d.departmentname, t.treatmentdate, t.cost

from treatments t

join patients p on t.patientid = p.patientid

join staff s on t.staffid = s.staffid

join departments d on s.departmentid = d.departmentid;

/\*

Subqueries and Correlated Subqueries

Use a subquery to find patients who have received treatments from a specific department (provided DepartmentID).

\*/

select distinct p.\*

from patients p

join treatments t on p.patientid = t.patientid

join staff s on t.staffid = s.staffid

where s.departmentid = 1;

-- Use a correlated subquery to find staff who have treated more than a specified number of patients (e.g., 3)

select \*

from staff s

where (select count(\*) from treatments t where t.staffid = s.staffid) > 2;

/\*

Aggregate Functions with GROUP BY and HAVING

Calculate the total cost of treatments per patient and filter out patients with a total treatment cost less than 1000

\*/

select p.name as patient\_name, sum(t.cost) as total\_cost

from patients p

join treatments t on p.patientid = t.patientid

group by p.name

having sum(t.cost) >= 1000;