HOSPITAL MANAGEMENT DB

ΠΑΝΑΓΙΩΤΗΣ ΚΕΠΕΣΙΔΗΣ – ΓΙΩΡΓΟΣ ΡΟΥΣΕΛΑΤΟΣ

Contents

Section 1: Introduction

This report provides a brief overview and explanation of the steps that were executed in order to create a mock hospital management database. It acts as a follow up to the first part of this assignment, where the concept was first presented, along with the relevant ER diagram and relational model. It is by no means meant to be an exhaustive analysis of the whole process, but more so an accompanying document aiming to aid the user of what the database is about, how it was created, how the data was loaded and how the SQL queries were exported.

Section 2: Creating the Hospital Management Database

2.a. Using the relational diagram tool of MySQL Workbench

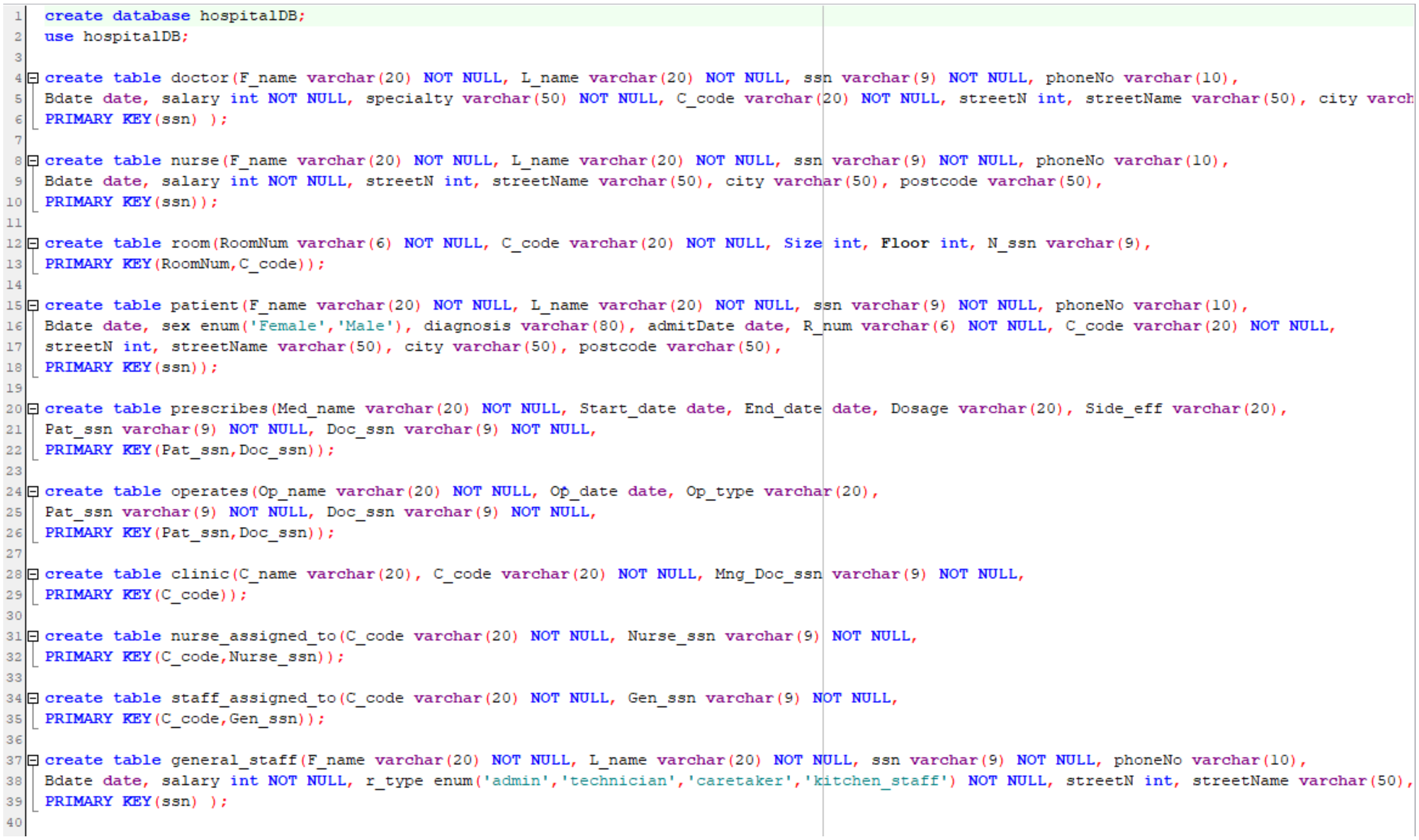
The database was created using statements written in pure SQL. However, in order to assist the creation process, the relational diagram generator of MySQL workbench tool was used to auto-generate the relational schema. The resulting relational diagram proved useful as a means of validation of table relationships and other crucial elements of the database.

2.b. Creating the tables and relationships

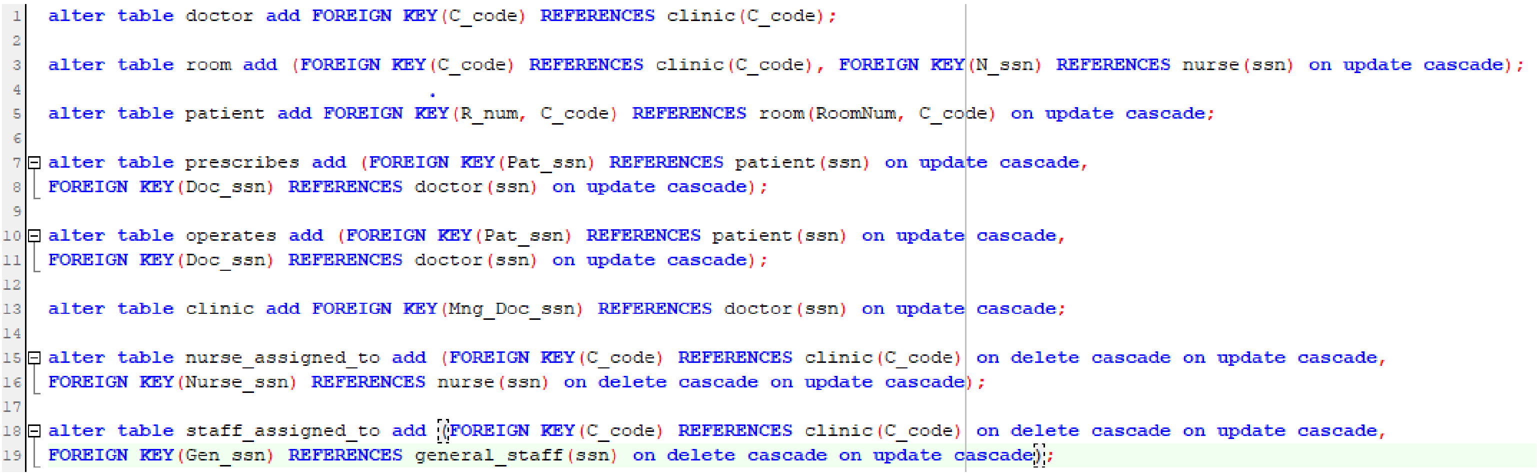
As mentioned already, queries written in MySQL were used in order to create and populate the database. This sub-section presents the creation part. The steps followed were:

- Writing the SQL statements to create the tables and the columns, including PK constraints.
- Adding the parts for creating FK and other constraints where such were necessary

The resulting statements are shown in the following screenshots:



```
create database hospitalDB;
use hospitalDB;

create table doctor(F_name varchar(20) NOT NULL, L_name varchar(20) NOT NULL, ssn varchar(9) NOT NULL, phoneNo varchar(10),
Bdate date, salary int NOT NULL, specialty varchar(50) NOT NULL, C_code varchar(20) NOT NULL, streetN int, streetName varchar(50), city varch
PRIMARY KEY(ssn) );

create table nurse(F_name varchar(20) NOT NULL, L_name varchar(20) NOT NULL, ssn varchar(9) NOT NULL, phoneNo varchar(10),
Bdate date, salary int NOT NULL, streetN int, streetName varchar(50), city varchar(50), postcode varchar(50),
PRIMARY KEY(ssn));

create table room(RoomNum varchar(6) NOT NULL, C_code varchar(20) NOT NULL, Size int, Floor int, N_ssn varchar(9),
PRIMARY KEY(RoomNum,C_code));

create table patient(F_name varchar(20) NOT NULL, L_name varchar(20) NOT NULL, ssn varchar(9) NOT NULL, phoneNo varchar(10),
Bdate date, sex enum('Female','Male'), diagnosis varchar(80), admitDate date, R_num varchar(6) NOT NULL, C_code varchar(20) NOT NULL,
streetN int, streetName varchar(50), city varchar(50), postcode varchar(50),
PRIMARY KEY(ssn));

create table prescribes(Med_name varchar(20) NOT NULL, Start_date date, End_date date, Dosage varchar(20), Side_eff varchar(20),
Pat_ssn varchar(9) NOT NULL, Doc_ssn varchar(9) NOT NULL,
PRIMARY KEY(Pat_ssn,Doc_ssn));

create table operates(Op_name varchar(20) NOT NULL, Op_date date, Op_type varchar(20),
Pat_ssn varchar(9) NOT NULL, Doc_ssn varchar(9) NOT NULL,
PRIMARY KEY(Pat_ssn,Doc_ssn));

create table clinic(C_name varchar(20), C_code varchar(20) NOT NULL, Mng_Doc_ssn varchar(9) NOT NULL,
PRIMARY KEY(C_code));

create table nurse_assigned_to(C_code varchar(20) NOT NULL, Nurse_ssn varchar(9) NOT NULL,
PRIMARY KEY(C_code,Nurse_ssn));

create table staff_assigned_to(C_code varchar(20) NOT NULL, Gen_ssn varchar(9) NOT NULL,
PRIMARY KEY(C_code,Gen_ssn));

create table general_staff(F_name varchar(20) NOT NULL, L_name varchar(20) NOT NULL, ssn varchar(9) NOT NULL, phoneNo varchar(10),
Bdate date, salary int NOT NULL, r_type enum('admin','technician','caretaker','kitchen_staff') NOT NULL, streetN int, streetName varchar(50),
PRIMARY KEY(ssn) );
```

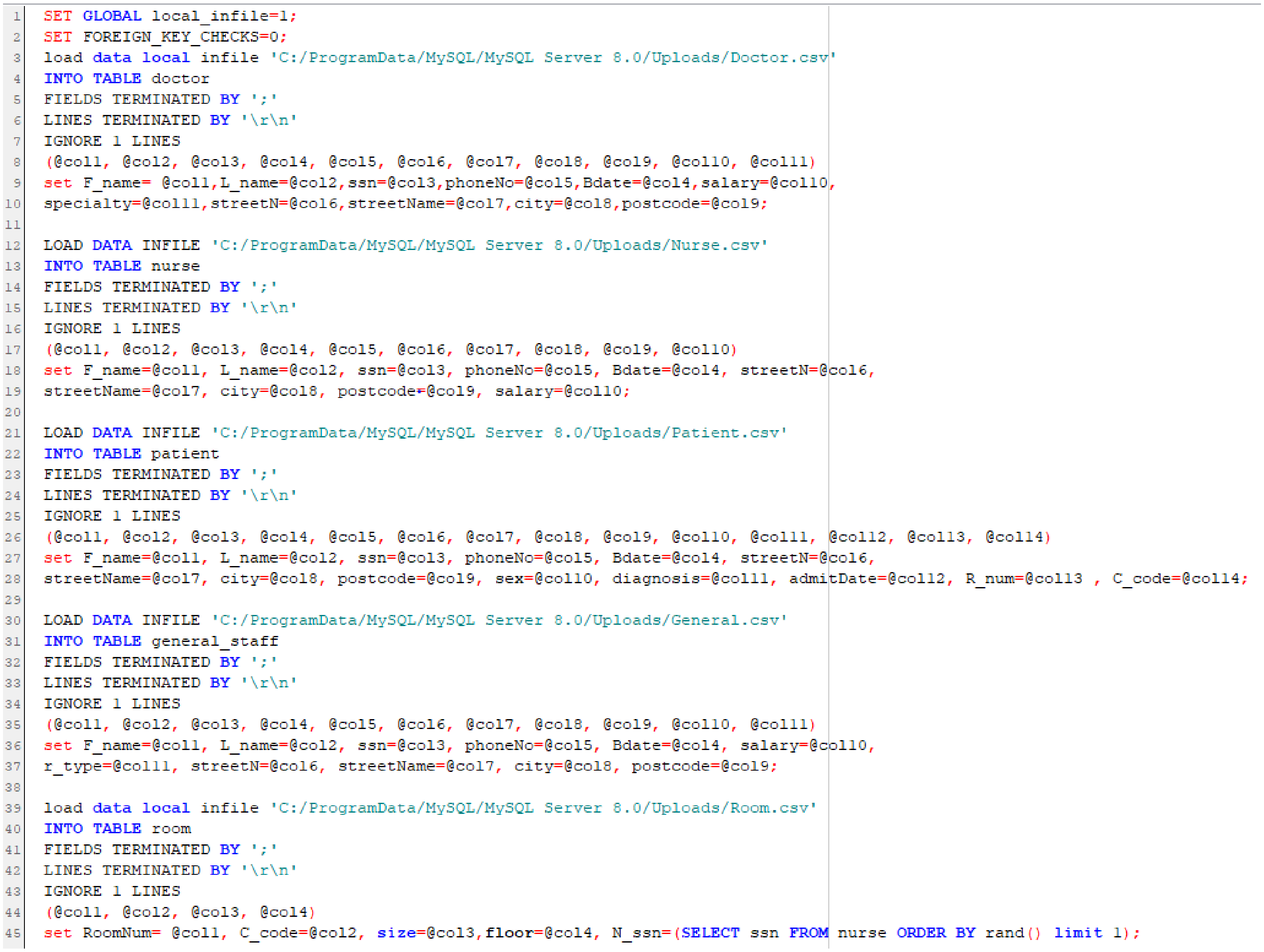


```
alter table doctor add FOREIGN KEY(C_code) REFERENCES clinic(C_code);

alter table room add (FOREIGN KEY(C_code) REFERENCES clinic(C_code), FOREIGN KEY(N_ssn) REFERENCES nurse(ssn) on update cascade);

alter table patient add FOREIGN KEY(R_num, C_code) REFERENCES room(RoomNum, C_code) on update cascade;

alter table prescribes add (FOREIGN KEY(Pat_ssn) REFERENCES patient(ssn) on update cascade,
FOREIGN KEY(Doc_ssn) REFERENCES doctor(ssn) on update cascade);

alter table operates add (FOREIGN KEY(Pat_ssn) REFERENCES patient(ssn) on update cascade,
FOREIGN KEY(Doc_ssn) REFERENCES doctor(ssn) on update cascade);

alter table clinic add FOREIGN KEY(Mng_Doc_ssn) REFERENCES doctor(ssn) on update cascade;

alter table nurse_assigned_to add (FOREIGN KEY(C_code) REFERENCES clinic(C_code) on delete cascade on update cascade,
FOREIGN KEY(Nurse_ssn) REFERENCES nurse(ssn) on delete cascade on update cascade);

alter table staff_assigned_to add (FOREIGN KEY(C_code) REFERENCES clinic(C_code) on delete cascade on update cascade,
FOREIGN KEY(Gen_ssn) REFERENCES general_staff(ssn) on delete cascade on update cascade);
```

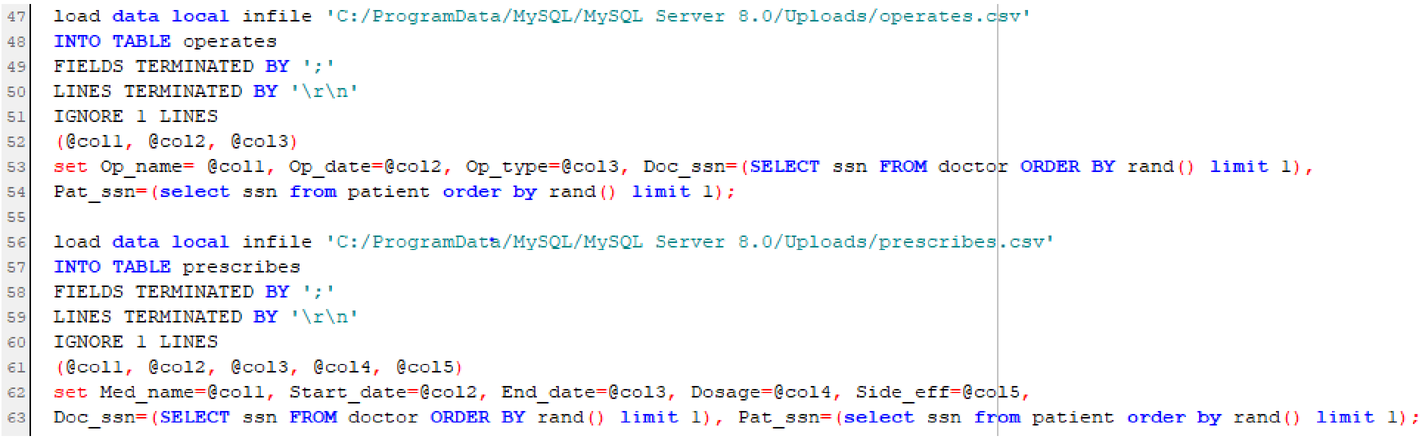
Section 3: Loading the Data

To populate the database, a couple of resources were used to create mock data which would be fitting to the schema. Those were a custom-made python program utilizing a library called Faker, which has several convenient constructors for the creation of random but valid data, based on whatever parameters the developer chooses. For more bespoke elements of data, such as medicine names (column in 'prescribes' table) for example, a random data generator website was used, called Mockaroo. Mockaroo allows the developer to get a dataset based on a broad variety of categories, ranging from health to manufacturing to aviation, etc.

The following screenshots show the queries used for the insertion of data in the database, reading them from .csv files:

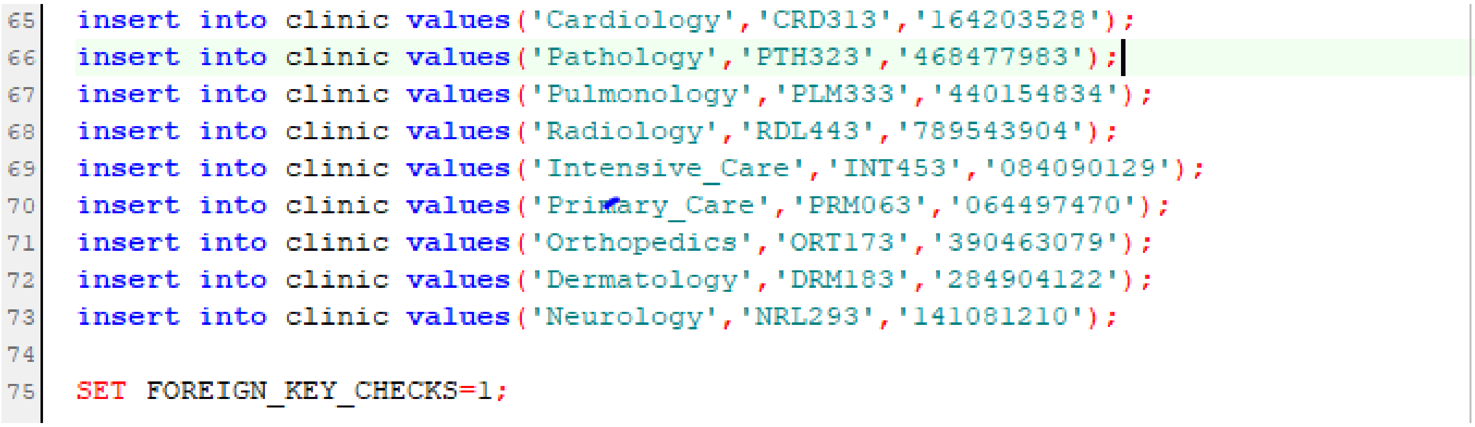


```sql
SET GLOBAL local_infile=1;
SET FOREIGN_KEY_CHECKS=0;
load data local infile 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Doctor.csv'
INTO TABLE doctor
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4, @col5, @col6, @col7, @col8, @col9, @col10, @col11)
set F_name= @col1,L_name=@col2,ssn=@col3,phoneNo=@col5,Bdate=@col4,salary=@col10,
specialty=@col11,streetN=@col6,streetName=@col7,city=@col8,postcode=@col9;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Nurse.csv'
INTO TABLE nurse
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4, @col5, @col6, @col7, @col8, @col9, @col10)
set F_name=@col1, L_name=@col2, ssn=@col3, phoneNo=@col5, Bdate=@col4, streetN=@col6,
streetName=@col7, city=@col8, postcode=@col9, salary=@col10;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Patient.csv'
INTO TABLE patient
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4, @col5, @col6, @col7, @col8, @col9, @col10, @col11, @col12, @col13, @col14)
set F_name=@col1, L_name=@col2, ssn=@col3, phoneNo=@col5, Bdate=@col4, streetN=@col6,
streetName=@col7, city=@col8, postcode=@col9, sex=@col10, diagnosis=@col11, admitDate=@col12, R_num=@col13 , C_code=@col14;

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/General.csv'
INTO TABLE general_staff
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4, @col5, @col6, @col7, @col8, @col9, @col10, @col11)
set F_name=@col1, L_name=@col2, ssn=@col3, phoneNo=@col5, Bdate=@col4, salary=@col10,
r_type=@col11, streetN=@col6, streetName=@col7, city=@col8, postcode=@col9;

load data local infile 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Room.csv'
INTO TABLE room
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4)
set RoomNum= @col1, C_code=@col2, size=@col3,floor=@col4, N_ssn=(SELECT ssn FROM nurse ORDER BY rand() limit 1);
```



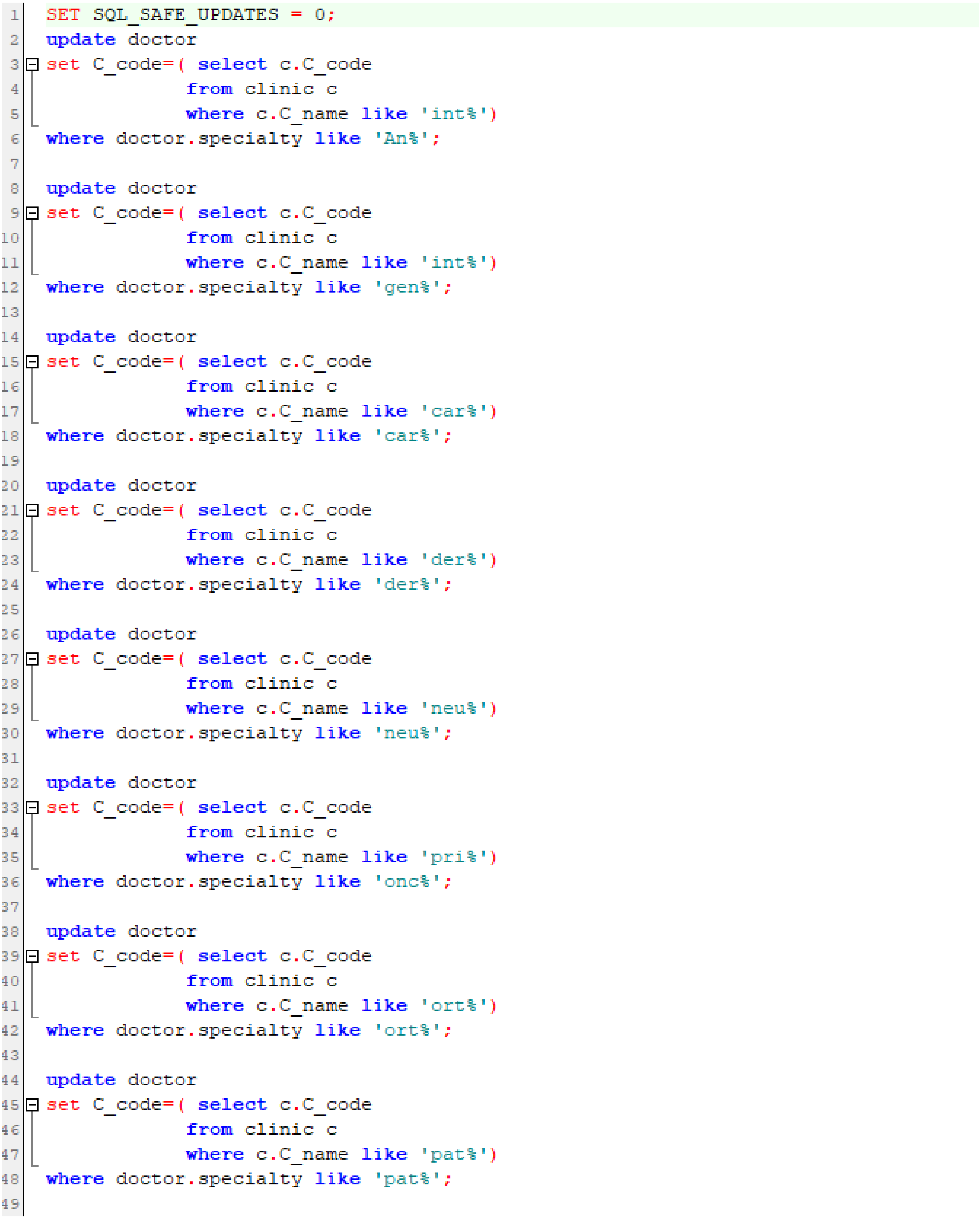
```
load data local infile 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/operates.csv'
INTO TABLE operates
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3)
set Op_name= @col1, Op_date=@col2, Op_type=@col3, Doc_ssn=(SELECT ssn FROM doctor ORDER BY rand() limit 1),
Pat_ssn=(select ssn from patient order by rand() limit 1);

load data local infile 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/prescribes.csv'
INTO TABLE prescribes
FIELDS TERMINATED BY ';'
LINES TERMINATED BY '\r\n'
IGNORE 1 LINES
(@col1, @col2, @col3, @col4, @col5)
set Med_name=@col1, Start_date=@col2, End_date=@col3, Dosage=@col4, Side_eff=@col5,
Doc_ssn=(SELECT ssn FROM doctor ORDER BY rand() limit 1), Pat_ssn=(select ssn from patient order by rand() limit 1);
```

For certain tables which were meant to only have a minimal number of data rows, the inserted data was embedded within the query:

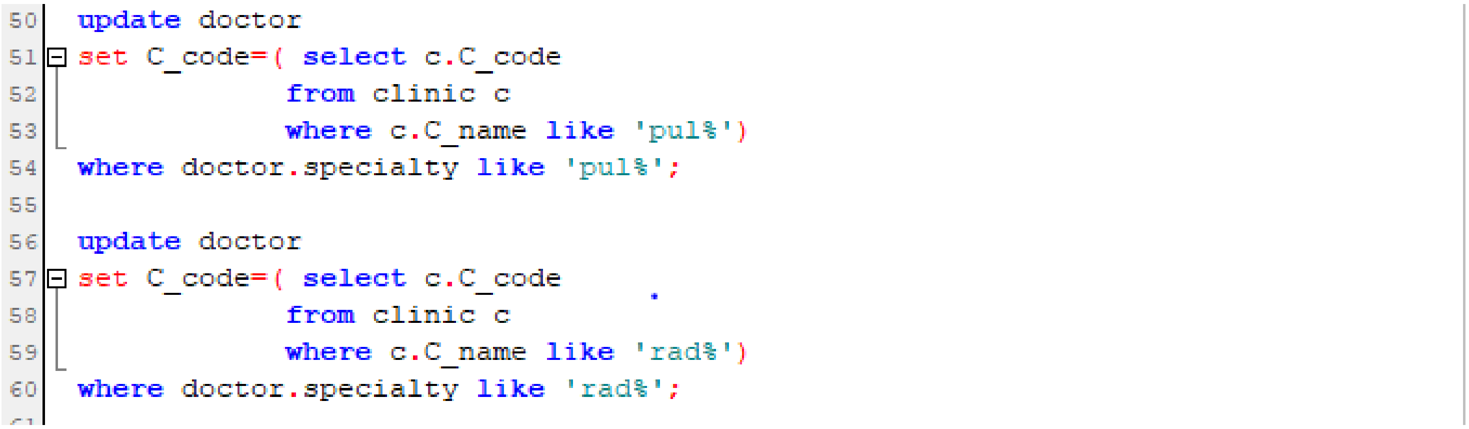


```
insert into clinic values('Cardiology','CRD313','164203528');
insert into clinic values('Pathology','PTH323','468477983');
insert into clinic values('Pulmonology','PLM333','440154834');
insert into clinic values('Radiology','RDL443','789543904');
insert into clinic values('Intensive_Care','INT453','084090129');
insert into clinic values('Primary_Care','PRM063','064497470');
insert into clinic values('Orthopedics','ORT173','390463079');
insert into clinic values('Dermatology','DRM183','284904122');
insert into clinic values('Neurology','NRL293','141081210');

SET FOREIGN_KEY_CHECKS=1;
```

There was also a case where an FK column in a table was populated by randomly, obviously observing any relational rules and constraints that were relevant to that table's column:

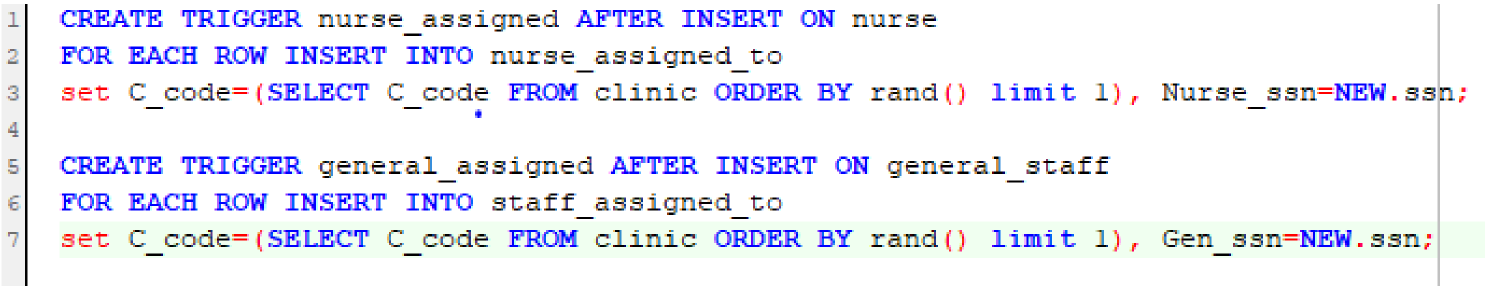


```
SET SQL_SAFE_UPDATES = 0;
update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'int%')
where doctor.specialty like 'An%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'int%')
where doctor.specialty like 'gen%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'car%')
where doctor.specialty like 'car%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'der%')
where doctor.specialty like 'der%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'neu%')
where doctor.specialty like 'neu%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'pri%')
where doctor.specialty like 'onc%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'ort%')
where doctor.specialty like 'ort%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'pat%')
where doctor.specialty like 'pat%';

```



```sql
update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'pul%')
where doctor.specialty like 'pul%';

update doctor
set C_code=( select c.C_code
            from clinic c
            where c.C_name like 'rad%')
where doctor.specialty like 'rad%';
```

Finally, to ensure the that the database stays valid according to its schema, triggers for inserting data were created wherever they were needed:



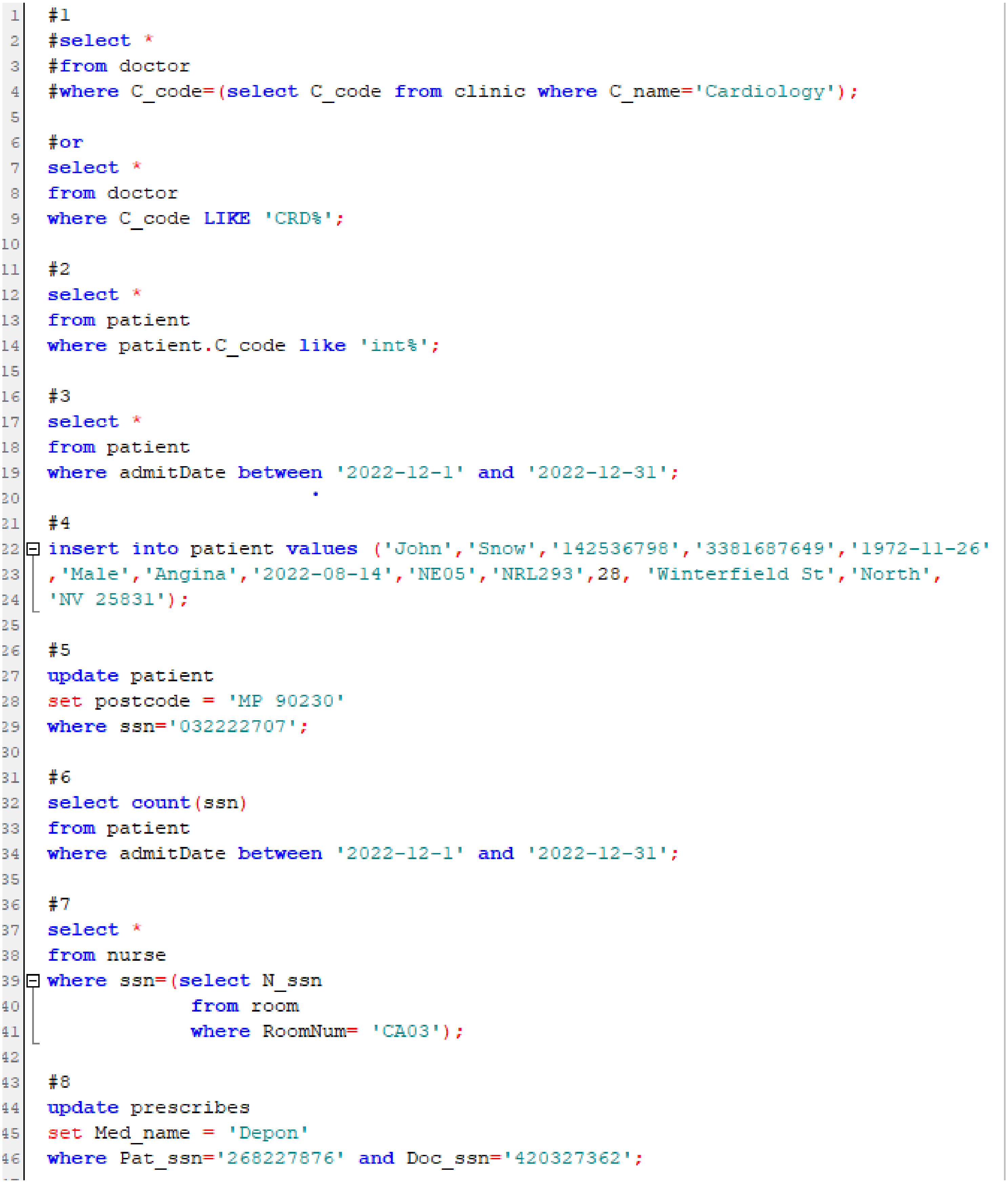
```sql
CREATE TRIGGER nurse_assigned AFTER INSERT ON nurse
FOR EACH ROW INSERT INTO nurse_assigned_to
set C_code=(SELECT C_code FROM clinic ORDER BY rand() limit 1), Nurse_ssn=NEW.ssn;

CREATE TRIGGER general_assigned AFTER INSERT ON general_staff
FOR EACH ROW INSERT INTO staff_assigned_to
set C_code=(SELECT C_code FROM clinic ORDER BY rand() limit 1), Gen_ssn=NEW.ssn;
```

Section 4: Exporting the SQL queries

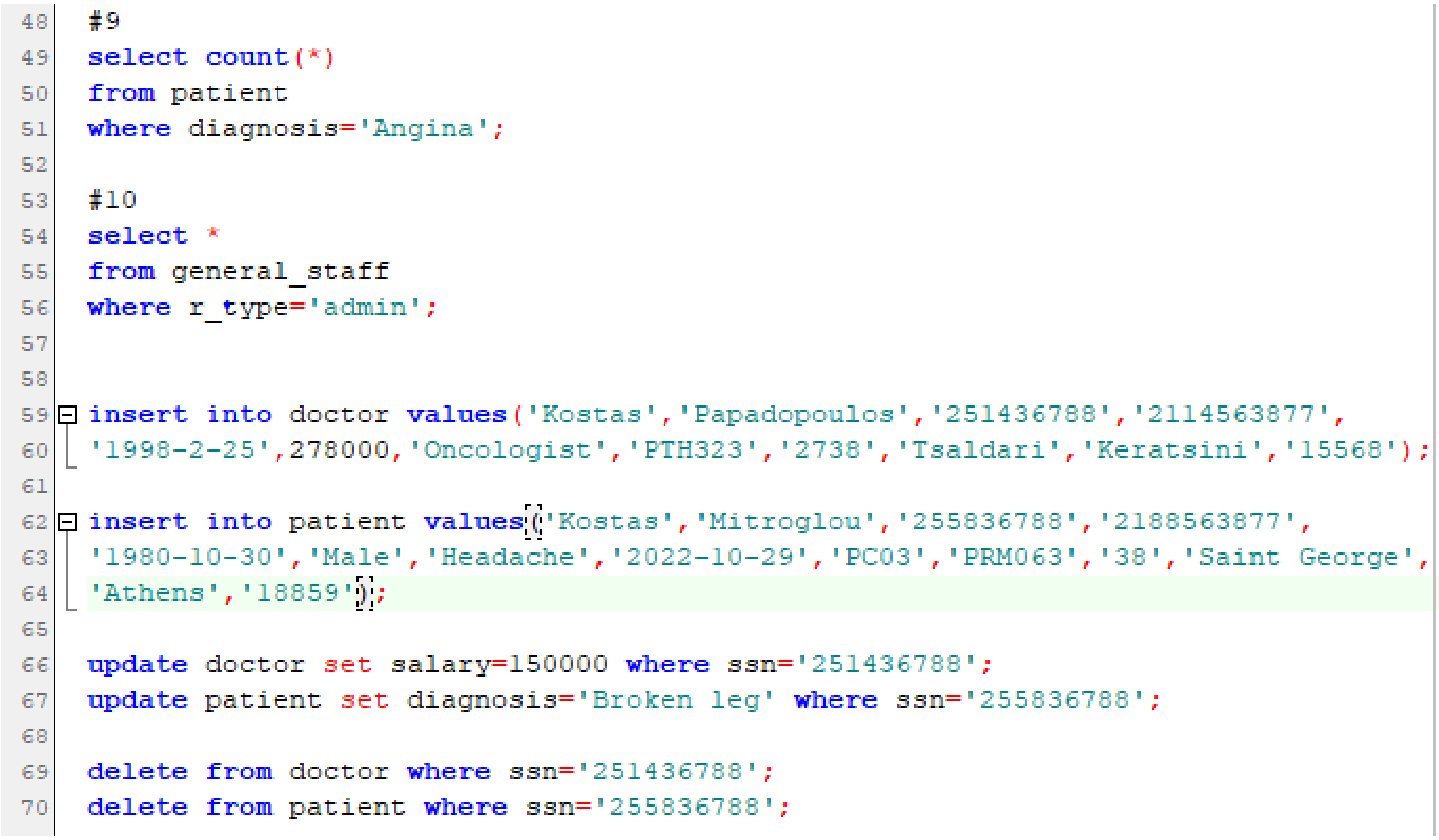
As per the assignment’s requirements, those queries that were written in plain English in part 1 were translated to SQL for the second part. As a reference, the statements in plain English from the first part:

- Bring all the doctor records from the cardiology (or any other) clinic.
- Bring all the patient records from the intensive care unit.
- Return all the records of patients that were admitted to the hospital within the last month.
- Insert a new patient record to a specific room.
- Update a patient record to input new correct postcode.
- Return the count of all patients that were admitted during last month.
- Bring the record of the nurse who is assigned to a specific room.
- Edit the medicine prescription of a given patient according to the supervising doctor’s instructions.
- Count all the patient records with a specific medical problem.
- Return all the admin staff (non-medical support staff) that work as desk administrators.

The above statements are illustrated in SQL in the screenshots below, with the addition of two insert statements, two update statements and two delete statements:



```
#1
#select *
#from doctor
#where C_code=(select C_code from clinic where C_name='Cardiology');

#or
select *
from doctor
where C_code LIKE 'CRD%';

#2
select *
from patient
where patient.C_code like 'int%';

#3
select *
from patient
where admitDate between '2022-12-1' and '2022-12-31';

#4
insert into patient values ('John','Snow','142536798','3381687649','1972-11-26'
,'Male','Angina','2022-08-14','NE05','NRL293',28, 'Winterfield St','North',
'NV 25831');

#5
update patient
set postcode = 'MP 90230'
where ssn='032222707';

#6
select count(ssn)
from patient
where admitDate between '2022-12-1' and '2022-12-31';

#7
select *
from nurse
where ssn=(select N_ssn
            from room
            where RoomNum= 'CA03');

#8
update prescribes
set Med_name = 'Depon'
where Pat_ssn='268227876' and Doc_ssn='420327362';
```



```
#9
select count(*)
from patient
where diagnosis='Angina';

#10
select *
from general_staff
where r_type='admin';


insert into doctor values('Kostas','Papadopoulos','251436788','2114563877',
'1998-2-25',278000,'Oncologist','PTH323','2738','Tsaldari','Keratsini','15568');

insert into patient values('Kostas','Mitroglou','255836788','2188563877',
'1980-10-30','Male','Headache','2022-10-29','PC03','PRM063','38','Saint George',
'Athens','18859');

update doctor set salary=150000 where ssn='251436788';
update patient set diagnosis='Broken leg' where ssn='255836788';

delete from doctor where ssn='251436788';
delete from patient where ssn='255836788';
```

END OF REPORT