Proposed System

The main aim of this project is to give information about the Functioning of Databases in Hospital System. A hospital database will allow creating and maintaining database including information of all the patients, all the hospital employee's including the doctors and the receptionist and the money transactions or the payment happening between the hospital and the patients which will be helpful in keeping a track of all sorts of transactions.

Patients entering the hospital will need to register first by filling a form at the reception. Each form has a unique number whose domain will be like all-natural numbers which will be assigned to that patient in the form of pat_id with the integer datatype and will be getting stored into the patient information database. This database will also include the purpose of visit and along with the name of the doctor they want to see. There will be database containing an information about the doctors within the hospital. Each doctor will be assigned a unique identification number whose domain will be like all-natural numbers. There will be another database containing information about all the employees working in the hospital. Each of these employees will be provided a unique identification number whose domain will be like all-natural numbers.

The user groups involved with respect to Hospital database are,

1. Receptionist
2. Doctors
3. Hospital Staff (Accountant)

Requirements:
Hospital database is a vast application and in real time, it involves many users and in this project, we will focus only on three users and their uses in the system.

Receptionist: The receptionist will be responsible for managing the appointments with the doctor. Also, the receptionist needs to keep track about the information of all the patients coming in and going out. Each patient who fills up the form will be assigned a particular unique id available on the form. Also, the receptionist can access the database to help the visitors by retrieving all the information about the patients and help them out in the best way. They can access the doctor information like the name of the doctor and their availability etc.

Doctors: The Doctor user group can view the database and review the activities going around with the patients. They need to see the lab reports of the patients in order to provide treatment accordingly. From the lab reports, the doctors can decide whether that particular patient need to get admitted or not. For that each patient and doctor should be uniquely defined and should follow the appropriate constraints. According to the type of the treatment given, the patient will be charged and given a room by the member of the staff working under that doctor.

Accountant: Accountants will be responsible for handling billing of patients going out or getting discharged. They will be accessing the database in order to check whether or what amount of

advances has been given by the patients and how much more they need to pay. They are also responsible for updating the billing info with the time. The patient will be charged accordingly depending upon the room or the treatment that has been given.

Overall Objective:

- Registration: Patients register by filling visiting form.
- Patients Details: Patient Bio data.
- Appointment: Reception set the appointment with the available doctor for the patients who wants to see the doctor for consultation. Other than those, the patients don't need to take the appointment specially for emergency cases.
- Doctors: Doctors Information + appointment
- Consultation: Doctor checks patient
- Lab report: Patient is either admitted or discharged as per the lab result.
- Staff: Hospital employee information
- Bill: Patient is finally billed out and get discharge.

Domain Requirements:

The domain requirements are as following:

i. All the patients, doctors as well as the hospital employees are assigned a unique identification number.
ii. All the unique number assigned are not nullable, i.e. they can't be assigned a null value.
iii. The minimum salary of any doctor should be $50000

Description:

To implement the above requirements, we would require a following relation.

1) The relation "Patientinfo" contains all the information about the patient visiting the hospital such as patient name, age, date of birth, gender, address etc. The patient id provided is unique to all patients. Each doc_id references to the doc_id mentioned in the Docinfo relation.

Format for dob is MM/DD/YYYY.

Domain for gender is M/F.

Primary key for this relation would be pat_id, date as the patient can come twice in the hospital for different purpose and for visiting different doctors. So, based upon the pat_id, date, the date value will differentiate the purpose of the patient visiting the hospital every time.

Patientinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
pat_id	integer		not null, unique	
pat_name	varchar	20	not null	
pat_fname	varchar	20	not null	

dob	date		not null	
gender	char	1	not null, (M,F)	
p_address	varchar	50	not null	
phoneno	varchar	10	not null, (All natural numbers)	
weight	integer		not null	
other_det	varchar	20		
doc_id	Integer		unique	delete, update
date	timestamp		not null, unique	

Primary Key: pat_id, date
Foreign Key: **doc_id**
(doc_id references doc_id in Docinfo)
Indices: doc_id

Attributes Details:

pat_id – Indicates the unique patient ID upon visiting the hospital
pat_name – Indicates the name of the patient
pat_fname – Indicates the First name of the patient
dob – Indicates the date of birth of the patient in the MM/DD/YYYY format.
gender – Indicates the gender of the patient
p_address – Indicates the address of the patient
phoneno – Indicates the contact number of the patient
weight – Indicates weight of the patient
other_det – Other details about the patient that he or she wants to describe.
doc_id – Indicates the Doctor id that the patient wants to visit or already visited.

date – Indicates the date time at which patient got registered.

2) The relation "Docinfo" holds the information of all the Doctors within the hospital. It includes the attributes like doc_id, doc_name, dept_name, gender, address etc.
Domain for gender is M/F.
Domain for designation is Jr/ Sr/ Surgeon/ Assistant/ Specialist.

Docinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
doc_id	integer		not null	
doc_name	varchar	20	not null	
doc_fname	varchar	20	not null	
gender	char	1	not null, (M,F)	
d_address	varchar	50	not null	
designation	varchar	10	not null, (Jr, Sr, Surgeon, Assistant, Specialist)	
salary	decimal	(10,2)	not null, Check(salary>50000)	

Primary Key: doc_id
Foreign Key:
Indices:

Attributes Details:

doc_id – Indicates the unique doctor ID which represents a particular doctor.
doc_name – Indicates the doctor full name
doc_fname - Indicates the first name of the doctor
gender – Indicates the gender of the doctor.
d_address – Indicates the address of the doctor.

designation – Indicates the designation of the doctor (Jr, Sr, Surgeon, Assistant, Specialist)
salary – Indicates the salary of the doctor.

3) The relation "Labinfo", corresponds to the information related to the lab reports of the patients or the disease diagnostic reports from which the doctor can conclude or provide treatment. It contains the lab_id which uniquely identifies the lab report, pat_id that references to patient that has been under diagnosis and doc_id that references to doctor under which this lab report has been initiated.

Labinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
lab_id	integer		not null	
pat_id	integer		not null	delete, update
weight	integer		not null	
doc_id	integer		not null	delete, update
labdate	timestamp		not null	
lab_charge	decimal	(8,2)	not null	
diag_details	varchar	50	not null	
remark	varchar	50		
other	varchar	100		

Primary Key: lab_id
Foreign Keys: **pat_id**
(pat_id references pat_id in Patinfo)
doc_id
(doc_id references doc_id in Docinfo)

Indices: pat_id, doc_id

Attributes Details:

lab_id – Indicates the unique lab ID of the lab report generated.
pat_id – Patient ID
weight – Weight of the patient as per the lab report
doc_id – Doctor ID
labdate - Date at which the lab report generated.
lab_charge – Charges of the lab report
diag_details – Diagnosis details that have been generated within the report about the patient
remark – Other remarks within the report
other – Other details

4) The relation "Staffinfo" holds the information about the employees working within the hospital. It holds a unique staff_id, dept_name in which that person is working and other info. This information can be grabbed by the doctor or the hospital administration.
doc_id is the attribute which contains the data about the staff that has been assigned to a particular doctor. The doc_id should be unique but it can contain null values as there could be some staff like accountant or receptionist who are independent employee in their department.
Domain for gender is M or F.
Here, one staff can be assigned to just one doctor or else none.
A staff can't be assigned to multiple doctors or vice versa.

Staffinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
staff_id	integer		not null	
s_Name	varchar	20	not null	
staff_fname	varchar	20	not null	
s_dept_name	varchar	20	not null	
gender	char	1	not null, (M,F)	

address	varchar	50	not null	
phoneno	varchar	10	not null	
doc_id	integer			delete, update

Primary Key: staff_id
Foreign Key: **doc_id**
(doc_id references doc_id in Docinfo)
Indices: doc_id

Attributes Details:

staff_id – Indicates the unique Staff ID
s_Name – Full Name of the staff
staff_fname – First name of the staff
s_dept_name – Department name in which the staff is working
gender – gender of the staff either (M or F)
address – Address of the staff
phoneno – Contact number of the staff
doc_id – Doctor ID under which staff is working or has been appointed to work for.

5) "Inpatient" relation contains all the details of the patients, who are admitted to the hospital after consulting from doctors by looking at the lab reports.
Lab_id in this relation is not unique. So, it can be inserted more than once. Also, if the patient change rooms during their stay, the room no will get change and therefore, the composite primary key defined over here will also get change and will be unique. Also, if the person is getting shifted back to the same room, then also it will be considered as unique as the data of admission relate to the date the patient is getting admitted into that room and the date of discharge is the date the patient is getting discharged from that room. So, the first date of admission and the last date of discharge will be considered as the date the patient is getting admitted into the hospital as well discharge from the hospital.

Inpatientinfo

Attribute Name	Data Type	Size	Constraints/Domains	Cascading Problems
pat_id	integer		not null	delete, update
room_no	integer		not null	delete, update
date_of_adm	timestamp		not null	
date_of_disch	timestamp		not null	
lab_id	integer		not null	delete, update

Primary Keys: pat_id, room_no, date_of_adm
Foreign Keys: **room_no**
(room_no references room_no in Roominfo)
pat_id
(pat_id references pat_id in Patinfo)
lab_id
(lab_id references lab_id in Labinfo)
Indices: lab_id

Attributes Details:

pat_id – Id of the patient getting admitted into the hospital
room_no – number of the room getting allotted to that patient
date_of_adm – date at which the patient is getting admitted
date_of_disch – date at which the patient is getting discharge
lab_id – id of the lab report for that patient

6) "Outpatient" relation contains all the details of the patients, who are not admitted to the hospital after consulting from doctors by looking at the lab reports.

Outpatientinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
pat_id	integer		not null	delete, update
date	timestamp		not null	

lab_id	integer		not null	delete, update

Primary Key: pat_id, lab_id
Foreign Keys: **pat_id**
(pat_id references pat_id in Patinfo)
lab_id
(lab_id references lab_id in Labinfo)
Indices: lab_id

Attributes Details:

pat_id – ID of the patient going out of the hospital without need of getting admitted.
date – date time at which the patient is going out
lab_id – id of the lab report for that particular patient

7) "Roominfo" relation contains all the details about the room. It includes attributes like room_no, room_type, total_beds, bed_occupied, status. It includes information like whether the room has been occupied or whether there are any empty beds available for the patient.
Domain for room_type is Single, Duplex, Triplex, Common.
Domain for Status is vacant or occupied.

Roominfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
room_no	integer		not null	
room_type	varchar	7	not null	
tot_beds	integer		not null	
no_bd_occupd	integer		not null	
status	varchar	8	not null, (vacant, occupied)	
floor_no	integer		not null	

Primary Key: room_no
Foreign Key:
Indices:

Attributes Details:

room_no – *Unique number allotted to the room within the hospital*
room_type – Type of the room (Single, Duplex, Triplex, Common)
tot_beds – Number of beds within a room
no_bd_occupd – Number of beds occupied within a room
status – Status of the Room (Whether it is vacant or occupied. So, if even one bed is empty in a room it will show as vacant. If all the beds in a room are occupied, then it will show as occupied.)
floor_no – floor number on which room is located.

8) "Billinfo" relation contains all the details about the billing of patients going out or patients getting discharge. It includes attributes like bill_id, pat_id, doc_charge etc.
For doc_charge, med_charge, room_charge, operation_charge, nursing_charge, lab_charge, no_of_days, advance, if nothing is entered then by default it will enter the value 0.

Billinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
bill_id	integer		not null	
pat_id	integer		not null	delete, update
pat_name	varchar	20	not null	
doc_name	varchar	20	not null	
doc_charge	integer		not null	
med_charge	integer		not null	
room_charge	integer		not null	
operation_charge	integer		not null	
no_of_days	integer		not null	
nursing_charge	integer		not null	

advance	integer		not null	
lab_charge	integer		not null	
datetime	timestamp		not null	

Primary Key: bill_id
Foreign Key: **pat_id**
(pat_id references pat_id in Patinfo)
Indices: pat_id

Attributes Details:

bill_id – Unique ID associated with each bill
pat_id – ID of the patient
pat_name – Full name of the Patient associated with that bill
doc_name – Full name of the doctor
doc_charge – Charge of the doctor
med_charge – Medical Charge
room_charge – Room charge if the patient is allocated any room.
operation_charge – Operation Charge
no_of_days – Number of days the patient stayed in hospital.
nursing_charge – Nursing Charge
advance – Any advance amount given by the patient
lab_charge – Charge for the Lab report
datetime – Date time at which the bill is created or updated.

9) "Schedinfo" relation contains all the details about the appointments that have been scheduled with the doctors for consultation by the receptionist.

Schedinfo				
Attribute Name	**Data Type**	**Size**	**Constraints/Domains**	**Cascading Problems**
app_id	integer		not null	
pat_id	integer		not null	delete, update
doc_id	integer		not null	delete, update
app_date	timestamp		not null	
datetime	timestamp		not null	

Primary Key: app_id, pat_id, doc_id
Foreign Keys: **pat_id**
(pat_id references pat_id in Patinfo)
doc_id
(doc_id references doc_id in Docinfo)

Attributes Details:

app_id – Unique number associated with each appointment
pat_id – ID of the patient for the appointment
doc_id – ID of the doctor for the appointment
app_date – date time at which the appointment is scheduled for
datetime – date time at which the appointment is created

This is the entire billing system which is integrated within a single database.

Note: I haven't considered all the situations related to the hospital management system, because there could be many more relations or attributes depending upon the hospital requirements.

Project Part 3: Normalization

1) UNF:

For our database to be in UNF, let us bring all the attributes from our relations into one relation, say Hospital. Hence our new relation in UNF looks like this.

[Note: I have changed some of the attributes in project 3 as compare to project 2 & 1, because I was facing difficult in naming the attributes which are similar in terms but serving different purpose like gender for patient and gender for doctor, So I tried naming this attributes as p_gender for patient and d_gender for doctor in order to differentiate. Also, I am removing advance attribute from the relation as its of no use which I got to know now. Changed attribute nursing_charging to staff_charge, s_dept_name to dept_name]

Hospital(pat_id, pat_name, pat_fname, dob, p_gender, p_street_address, p_city, p_state, p_country, p_postal_code, p_phoneno, p_weight, other_det, p_date, doc_id, doc_name, doc_fname, d_gender, d_street_address, d_city, d_state, d_country, d_postal_code, d_phoneno, designation, salary, lab_id, l_weight, labdate, lab_charge, diag_details, remark, other, staff_id, s_Name, staff_fname, dept_name, s_gender, s_street_address, s_city, s_state, s_country, s_postal_code, s_phoneno, room_no, date_of_adm, date_of_disch, o_pat_date, room_type, tot_beds, no_bd_occupd, status, floor_no, bill_id, doc_charge, med_charge, room_charge, operation_charge, no_of_days, staff_charge, b_datetime, app_id, app_date, app_datetime)

2) 1NF:

For our database to be in 1NF, it is necessary for us to follow the following rules:

1. There must not be any multivalued attribute.
2. There must be no repetition.
3. Primary key must be defined.

The database follows the first two rules, but we need to assign a primary key. The set of primary keys that define the database are

- **pat_id**
- **doc_id**
- **lab_id**
- **date_of_adm**
- **staff_id**
- **room_no**
- **o_pat_date**
- **bill_id**
- **app_id**

Also, together these are unique. Hence, now our database looks like:

Hospital (pat_id, pat_name, pat_fname, dob, p_gender, p_street_address, p_city, p_state, p_country, p_postal_code, p_phoneno, p_weight, other_det, p_date, doc_id, doc_name, doc_fname, d_gender d_street_address, d_city, d_state, d_country, d_postal_code, d_phoneno, designation, salary, lab_id, l_weight, labdate, lab_charge, diag_details, remark, other, staff_id, s_Name, staff_fname, dept_name, s_gender, s_street_address, s_city, s_state, s_country, s_postal_code, s_phoneno, room_no, date_of_adm, date_of_disch, o_pat_date, room_type, tot_beds, no_bd_occupd, status, floor_no, bill_id, doc_charge, med_charge, room_charge, operation_charge, no_of_days, staff_charge, b_datetime, app_id, app_date, app_datetime)

3) 2NF:

For our database to be in 2NF, we need to make sure that all non-key fields must depend on all the primary keys. For our table to be in 2NF we need to make the following changes:

1) Patient Information depends upon the patient ID that will be getting assigned to them while registration. Therefore, pat_name, pat_fname, dob, p_gender, p_street_address, p_city, p_state, p_country, p_postal_code, p_phoneno, p_weight, other_det, p_date depends upon the pat_id.

2) Doctor Information depends upon the Doctor ID and not on other primary keys. Therefore, doc_name, doc_fname, d_gender, d_street_address, d_city, d_state, d_country, d_postal_code, d_phoneno, designation, salary, doc_charge depends upon doc_id.

3) Then l_weight, labdate, diag_details, remark, lab_charge, other depends upon lab_id.

4) Then s_Name, staff_fname, dept_name, s_gender, s_street_address, s_city, s_state, s_country, s_postal_code, s_phoneno, staff_charge depends upon staff_id.

5) Then date_of_disch, no_of_days depends upon pat_id, room_no, date_of_adm.

6) Then room_type, tot_beds, no_bd_occupd, status, floor_no, room_charge depends upon room_no.

7) doc_charge, staff_charge, lab_charge, med_charge, operation_charge, room_charge, b_datetime depends upon bill_id

8) Then app_date, app_datetime depends upon app_id, doc_id and pat_id.

9) lab_date depends upon the lab_id, pat_id, doc_id as this is the junction table of the Patient, Doctor and Lab information relation.

Hence the new relation schema is:

Patientinfo (pat_id, pat_name, pat_fname, dob, p_gender, p_street_address, p_city, p_state, p_country, p_postal_code, p_phoneno, p_weight, other_det, p_date)

Doctorinfo (doc_id, doc_name, doc_fname, d_gender, d_street_address, d_city, d_state, d_country, d_postal_code, d_phoneno, designation, salary, doc_charge)

Doc_Staff (doc_id, staff_id) [Doc_Staff relation is the relation which indicates the staff which has been appointed as a helper to the doctor. So, each doctor can have just one helper for example, nurse or ward boy and each doctor will be having a helper]

Labinfo (lab_id, l_weight, labdate, diag_details, remark, other, lab_charge)

Assist (lab_id, pat_id, doc_id, labdate)

Staffinfo (staff_id, s_Name, staff_fname, dept_name, s_gender, s_gender, s_street_address, s_city, s_state, s_country, s_postal_code, s_phoneno, staff_charge)

Inpatientinfo (room_no, pat_id, date_of_adm, date_of_disch, no_of_days)

Outpatientinfo (pat_id, o_pat_date)

Roominfo (room_no, room_type, tot_beds, no_bd_occupd, status, floor_no, room_charge)

Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge, operation_charge, room_charge, b_datetime)

Bill_Pat (bill_id, pat_id)

Schedinfo (app_id, pat_id, doc_id, app_date, app_datetime)

4) 3NF:

For our database to be in 3NF, we need to make sure that, no non-key value depends upon one another.

Hence, these are the following changes that we must make:

1) As salary and doc_charge is based upon the designation of the doctor, we need to separate it out and need to create another relation.

2) Also as room charge is based upon the type of the room, we need to separate it out and need to create another relation.

3) Also as staff charge is based upon the person working in a department, so we need to separate it out and need to create another relation.

Hence the relation schema looks like this:

Patientinfo (pat_id, pat_name, pat_fname, dob, p_gender, p_street_address, p_city, p_state, p_country, p_postal_code, p_phoneno, p_weight, other_det, p_date)

Doctorinfo (doc_id, doc_name, doc_fname, d_gender, d_street_address, d_city, d_state, d_country, d_postal_code, d_phoneno, designation)

Dr_design (designation, salary, doc_charge)

Doc_Staff (doc_id, staff_id)

Labinfo (lab_id, l_weight, labdate, diag_details, remark, other, lab_charge)

Assist (lab_id, pat_id, doc_id, labdate)

Staffinfo (staff_id, s_Name, staff_fname, dept_name, s_gender, s_street_address, s_city, s_state, s_country, s_postal_code, s_phoneno)

Staff_dept_charge (dept_name, staff_charge)

Inpatientinfo (room_no, pat_id, date_of_adm, date_of_disch, no_of_days)

Outpatientinfo (pat_id, o_pat_date)

Roominfo (room_no, room_type, tot_beds, no_bd_occupd, status, floor_no)

Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge, operation_charge, room_charge, b_datetime)

Roomcharge (room_type, room_charge)

Bill_Pat (bill_id, pat_id)

Schedinfo (app_id, pat_id, doc_id, app_date, app_datetime)

Now, we can see all the above-mentioned relations are in 3NF.

Note:

1) After normalization, I found out that Billinfo table needs to be calculated by joining other tables. So, I divided all the charges like medical charges, operation charges, doctor charges etc separately which I thought it would be correct way of doing as per the rules of normalization.

2) Also, primary keys for some of the relations are now different after applying normalization as compare to earlier.

SQL QUERIES:

```
Create table Patientinfo
(
pat_id integer,
pat_name varchar(20) not null,
pat_fname varchar(20) not null,
dob date not null,
p_gender char(1) not null check(p_gender in ('M','F')),
p_street_address varchar(30) not null,
p_city varchar(15) not null,
p_state varchar(15) not null,
p_country varchar(15) not null,
p_postal_code varchar(8) not null,
p_phoneno varchar(10) not null,
p_weight integer not null,
other_det varchar(30),
p_date timestamp not null,
primary key(pat_id)
);

create index pat_date on Patientinfo(p_date);
```

```
Create table Dr_design
(
designation varchar(10),
salary decimal(10,2) not null check(salary>50000),
doc_charge integer not null,
primary key(designation)
);

Create table Doctorinfo
(
doc_id integer,
doc_name varchar(20) not null,
doc_fname varchar(20) not null,
d_gender char(1) not null check(d_gender in ('M','F')),
d_street_address varchar(30) not null,
d_city varchar(15) not null,
d_state varchar(15) not null,
d_country varchar(15) not null,
d_postal_code varchar(8) not null,
d_phoneno varchar(10) not null,
designation varchar(10) not null check(designation in
('Jr','Sr','Surgeon','Assistant','Specialist')),
primary key(doc_id),
foreign key(designation) references Dr_design(designation) on delete cascade on update
cascade
);

create index design on Doctorinfo(designation);
```

```
Create table Staff_dept_charge
(
dept_name varchar(10) not null,
staff_charge integer not null,
primary key(dept_name)
);

Create table Staffinfo
(
staff_id integer,
s_Name varchar(20) not null,
staff_fname varchar(20) not null,
dept_name varchar(10) not null,
s_gender char(1) not null check(s_gender in ('M','F')),
s_street_address varchar(30) not null ,
s_city varchar(15) not null,
s_state varchar(15) not null,
s_country varchar(15) not null,
s_postal_code varchar(8) not null,
s_phoneno varchar(10) not null,
primary key(staff_id),
foreign key(dept_name) references Staff_dept_charge(dept_name) on delete cascade on
update cascade
);
```

```
Create table Doc_Staff
(
doc_id integer not null,
staff_id integer not null,
primary key(doc_id,staff_id),
foreign key(doc_id) references Doctorinfo(doc_id) on delete cascade,
foreign key (staff_id) references Staffinfo(staff_id) on delete cascade
);

Create table Labinfo
(
lab_id integer,
l_weight integer not null,
lab_date timestamp not null unique,
diag_details varchar(50) not null,
remark varchar(50),
other varchar(100),
lab_charge integer not null,
primary key(lab_id)
);
```

```
Create table Assist
(
lab_id integer not null,
pat_id integer not null,
doc_id integer not null,
lab_date timestamp not null,
primary key(lab_id,pat_id,doc_id),
foreign key(lab_id) references Labinfo(lab_id) on delete cascade,
foreign key(doc_id) references Doctorinfo(doc_id) on delete cascade,
foreign key(pat_id) references Patientinfo(pat_id) on delete cascade,
foreign key(lab_date) references Labinfo(lab_date) on delete cascade
);
```

```
Create table Roomcharge
(
room_type varchar(7),
room_charge integer not null,
primary key(room_type)
);
```

```
Create table Roominfo
(
room_no integer,
room_type varchar(7) not null check(room_type in ('Single','Duplex','Triplex','Common')),
tot_beds integer not null,
no_bd_occupd integer not null,
status varchar(8) not null check(status in ('vacant','occupied')),
floor_no integer not null,
primary key(room_no),
foreign key(room_type) references Roomcharge(room_type) on delete cascade on update
cascade
);

create index room_typ on Roominfo(room_type);
create index stats on Roominfo(status);

Create table Inpatientinfo
(
room_no integer not null,
pat_id integer not null,
date_of_adm timestamp not null,
date_of_disch timestamp not null,
no_of_days integer not null,
primary key(room_no, pat_id, date_of_adm),
foreign key(room_no) references Roominfo(room_no) on delete cascade,
foreign key(pat_id) references Patientinfo(pat_id) on delete cascade
);
```

```
Create table Outpatientinfo
(
pat_id integer not null,
o_pat_date timestamp not null,
primary key(pat_id,o_pat_date),
foreign key(pat_id) references Patientinfo(pat_id) on delete cascade on update cascade
);
```

```
Create table Medicalcharge
(
bill_id integer,
doc_charge integer not null,
lab_charge integer not null,
staff_charge integer not null,
med_charge integer default 0,
operation_charge integer default 0,
room_charge integer not null,
b_datetime timestamp not null,
primary key(bill_id)
);

create index bill_dt on Medicalcharge(b_datetime);
```

```
Create table Bill_Pat
(
bill_id integer not null,
pat_id integer not null,
primary key(bill_id,pat_id),
foreign key(bill_id) references Medicalcharge(bill_id) on delete cascade,
foreign key(pat_id) references Patientinfo(pat_id) on delete cascade
);

Create table Schedinfo
(
app_id integer,
pat_id integer not null,
doc_id integer not null,
app_date timestamp not null,
app_datetime timestamp not null,
primary key(app_id,pat_id,doc_id),
foreign key(pat_id) references Patientinfo(pat_id) on delete cascade,
foreign key(doc_id) references Doctorinfo(doc_id) on delete cascade
);
```

Inserting values into the relations:

insert into Patientinfo values(10001,'Arun Sharma','Arun','01/22/1991','M','A6 Spruce Street','Jersey City','NJ','US','07307',9876345612,65,'Suffering from cold and cough','06-12-2012 14:00:00');

insert into Patientinfo values(10002,'Vijay Sinha','Vijay','04/12/1986','M','B9 Prospect Street','Union City','NJ','US','07306',9876275612,70,'High Fever','06-08-2013 15:00:00');

insert into Patientinfo values(10003,'John Cooper','John','06/15/1978','M','C6 Thorne Street','Hoboken','NJ','US','0856',9876782612,78,'Throat pain','02-10-2014 18:00:00');

insert into Patientinfo values(10004,'Michelle Stanley','Michelle','08/09/1998','F','A4 Laven Street','Jersey City','NJ','US','07307',9901345612,60,'Stomach ache','05-12-2015 10:00:00');

insert into Patientinfo values(10005,'Ruby Yul','Ruby','03/25/1996','F','A6 Enble Street','Union City','NJ','US','03307',3467345612,60,'Heart ache','03-08-2016 11:00:00');

insert into Patientinfo values(10006,'Hanny Khub','Honey','06/20/1985','F','202 Enble Street','Union City','NJ','US','03307',3423445612,72,'Head ache','05-10-2016 11:00:00');

insert into Patientinfo values(10007,'Sameul Honey','Sameul','07/20/1988','M','201 Enle Street','Union City','NJ','US','03307',3421235612,63,'Head ache','05-20-2016 11:00:00');

insert into Doctorinfo values(101,'Stan Murphy','Stan','M','212 Hopkins Street','Hoboken','NJ','US',08234,7623569801,'Jr');

insert into Doctorinfo values(102,'Alice Yum','Alice','F','13 Hopkins Street','Hoboken','NJ','US',08454,7456569801,'Sr');

insert into Doctorinfo values(103,'Steve Jan','Steve','M','2 Ins Street','Union City','NJ','US',02334,7698013471,'Assistant');

insert into Doctorinfo values(104,'Mike Lanny','Mike','M','224 Central Street','Jersey City','NJ','US',08267,7690169801,'Surgeon');

insert into Doctorinfo values(105,'Camy Hun','Camy','F','2 Hopkins Street','Hoboken','NJ','US',08234,7623123801,'Specialist');

```
insert into Dr_design values('Jr',100000,1000);

insert into Dr_design values('Sr',200000,2000);

insert into Dr_design values('Assistant',300000,3000);

insert into Dr_design values('Surgeon',400000,4000);

insert into Dr_design values('Specialist',500000,5000);


insert into Doc_staff values(101,202);

insert into Doc_staff values(102,201);

insert into Doc_staff values(103,204);

insert into Doc_staff values(104,203);

insert into Doc_staff values(105,205);


insert into Labinfo values(1,70,'06-12-2012 18:00:00','Few Symptoms of Malaria','More tests
required',450);

insert into Labinfo values(2,75,'06-08-2013 19:00:00','Malaria','No more tests required',450);

insert into Labinfo values(3,80,'02-10-2014 20:00:00','Tonsils','More tests required',300);

insert into Labinfo values(4,61,'05-12-2015 13:00:00','Symptoms for Cancer','More tests
required',650);

insert into Labinfo values(5,65,'03-08-2016 14:00:00','Heart blockage','More tests
required',650);

insert into Labinfo values(6,73,'05-10-2016 13:00:00','Head ache','No more tests
required',450);


insert into Assist values(1,10001,102,'06-12-2012 18:00:00');

insert into Assist values(2,10002,101,'06-08-2013 19:00:00');

insert into Assist values(3,10003,102,'02-10-2014 20:00:00');

insert into Assist values(4,10004,105,'05-12-2015 13:00:00');

insert into Assist values(5,10005,104,'03-08-2016 14:00:00');

insert into Assist values(6,10006,101,'05-10-2016 13:00:00');
```

```
insert into Staffinfo values(201,'Sheela Trui','Sheela','Nurse','F','A2 Labr Street','Hoboken','NJ','US','78342',6527839789);

insert into Staffinfo values(202,'Manish Lui','Manish','Wardboy','M','B6 Lincholn Street','Hoboken','NJ','US','78342',4563839789);

insert into Staffinfo values(203,'Jenifer Yui','Jenifer','Nurse','F','L1 Central Street','Jersey City','NJ','US','07307',6528912789);

insert into Staffinfo values(204,'Vitram Hanny','Vitram','Wardboy','M','B8 Labr Street','Hoboken','NJ','US','78342',6521234789);

insert into Staffinfo values(205,'Junn Ham','Junn','Nurse','F','B9 Hopkins Street','Union City','NJ','US','34342',8977839789);

insert into Staffinfo values(206,'Judy Hum','Judy','Accountant','F','212 Hopkins Street','Union City','NJ','US','32342',8990459789);


insert into Staff_dept_charge values('Nurse',100);

insert into Staff_dept_charge values('Accountant',200);

insert into Staff_dept_charge values('Wardboy',80);

insert into Staff_dept_charge values('Cleaning',50);


insert into Inpatientinfo values(001,10001,'06-13-2012 10:00:00','07-12-2012 14:00:00',30);

insert into Inpatientinfo values(001,10002,'06-09-2013 15:00:00','07-08-2013 14:00:00',30);

insert into Inpatientinfo values(003,10003,'02-11-2014 18:00:00','02-13-2014 14:00:00',2);

insert into Inpatientinfo values(002,10004,'05-13-2015 10:00:00','05-12-2016 14:00:00',365);

insert into Inpatientinfo values(003,10005,'03-09-2016 11:00:00','04-08-2016 18:00:00',30);


insert into Outpatientinfo values(10006,'05-10-2016 15:00:00');


insert into Roominfo values(001,'Duplex',2,2,'occupied',2);

insert into Roominfo values(002,'Single',1,1,'occupied',2);

insert into Roominfo values(003,'Duplex',2,2,'occupied',2);
```

```
insert into Roominfo values(004,'Triplex',3,0,'vacant',1);

insert into Roominfo values(005,'Common',8,0,'vacant',1);


insert into Roomcharge values('Single',10000);

insert into Roomcharge values('Duplex',5000);

insert into Roomcharge values('Triplex',1000);

insert into Roomcharge values('Common',500);


insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
room_charge, b_datetime) values(1,2000,450,100,1000,5000,'07-12-2012 14:00:00');

insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
room_charge, b_datetime) values(2,1000,450,80,1000,5000,'07-08-2013 14:00:000');

insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
room_charge, b_datetime) values(3,2000,300,100,1000,5000,'02-13-2014 14:00:00');

insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
operation_charge, room_charge, b_datetime) values(4,5000,650,100,2000, 50000,10000,'05-
12-2016 14:00:00');

insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
operation_charge, room_charge, b_datetime) values(5,4000,650,100,2000,100000, 5000,'04-
08-2016 18:00:00');

insert into Medicalcharge (bill_id, doc_charge, lab_charge, staff_charge, med_charge,
room_charge, b_datetime) values(6,1000,450,80,1000,5000,'05-10-2016 15:00:00');


insert into Bill_Pat values(1,10001);

insert into Bill_Pat values(2,10002);

insert into Bill_Pat values(3,10003);

insert into Bill_Pat values(4,10004);

insert into Bill_Pat values(5,10005);

insert into Bill_Pat values(6,10006);
```

insert into Schedinfo values(001,10001,102,'06-12-2012 15:00:00','06-11-2012 14:00:00');

insert into Schedinfo values(002,10002,101,'06-08-2013 16:00:00','06-07-2013 15:00:00');

insert into Schedinfo values(003,10003,102,'02-10-2014 19:00:00','02-08-2014 18:00:00');

insert into Schedinfo values(004,10004,105,'05-12-2015 11:00:000','05-11-2015 10:00:00');

insert into Schedinfo values(005,10006,101,'05-10-2016 13:00:00','05-09-2016 11:00:00');

insert into Schedinfo values(006,10007,101,'05-21-2016 13:00:00','05-20-2016 11:00:00');

User 1-For Receptionist:

Query 1

1) List the number of appointments did Dr Camy had on 11th May 2015?

Select count(*)

from Schedinfo

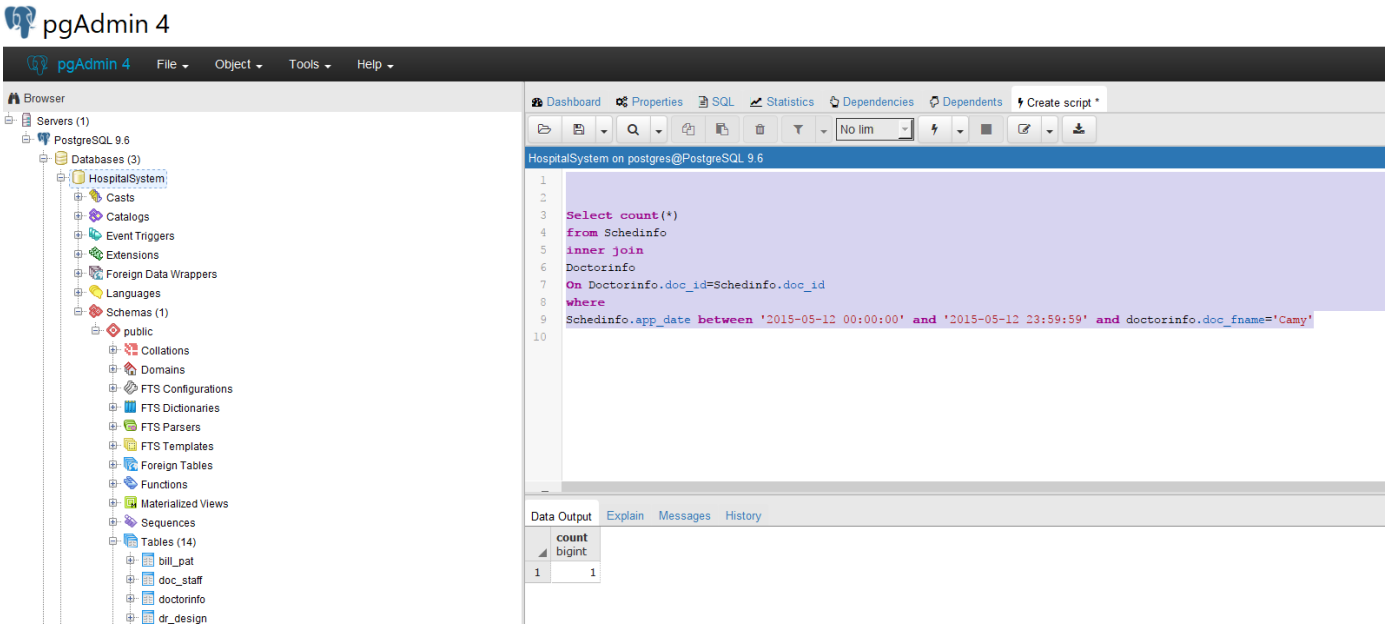
inner join

Doctorinfo

On Doctorinfo.doc_id=Schedinfo.doc_id

where

Schedinfo.app_date between '2015-05-12 00:00:00' and '2015-05-12 23:59:59' and doctorinfo.doc_fname='Camy'



Query 2

2) List the name of the patients who have been treated by Dr Stan?

Select a.pat_fname

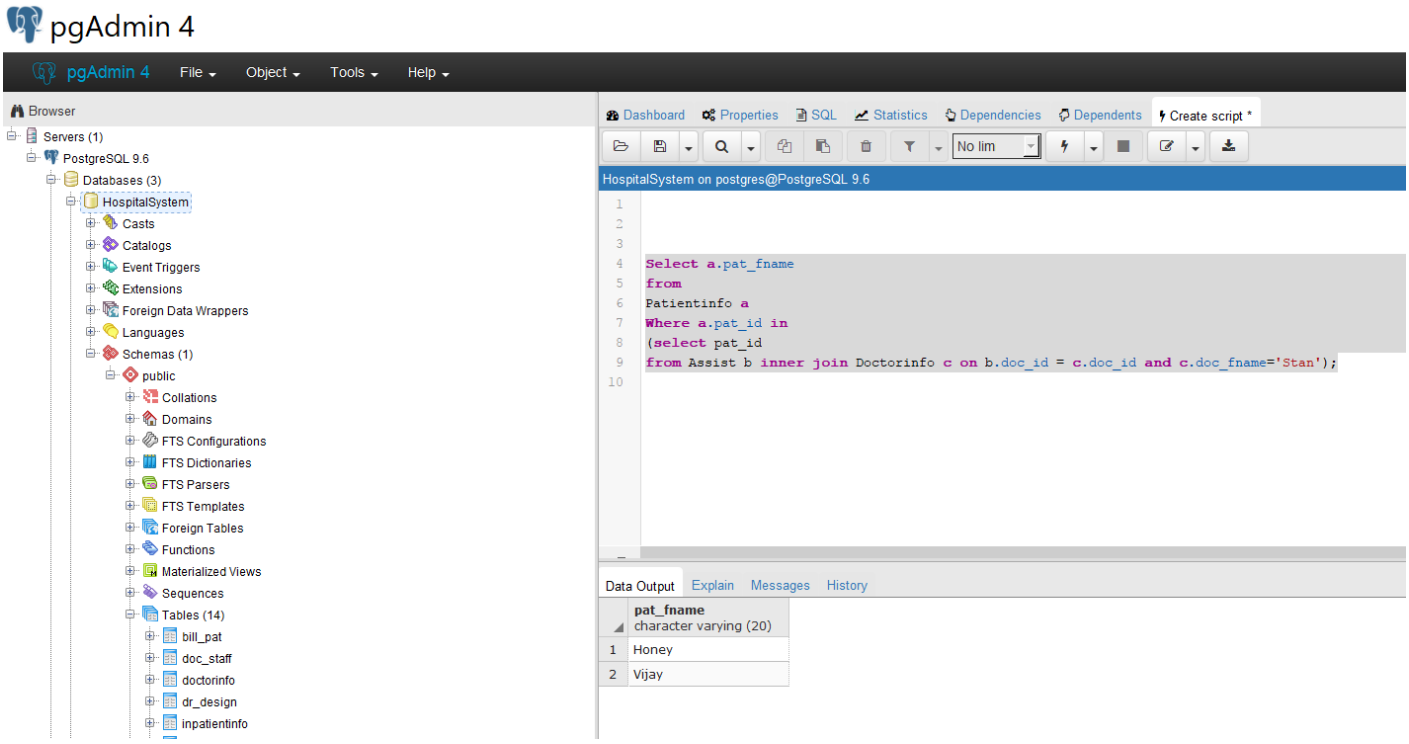
from

Patientinfo a

Where a.pat_id in

(select pat_id

from Assist b inner join Doctorinfo c on b.doc_id = c.doc_id and c.doc_fname='Stan');



Query 3

3) List all the patients who are admitted in room no 001 and having Malaria?

select Patientinfo.pat_fname

from

Patientinfo inner join Inpatientinfo on Patientinfo.pat_id=Inpatientinfo.pat_id

where Inpatientinfo.room_no=001 and

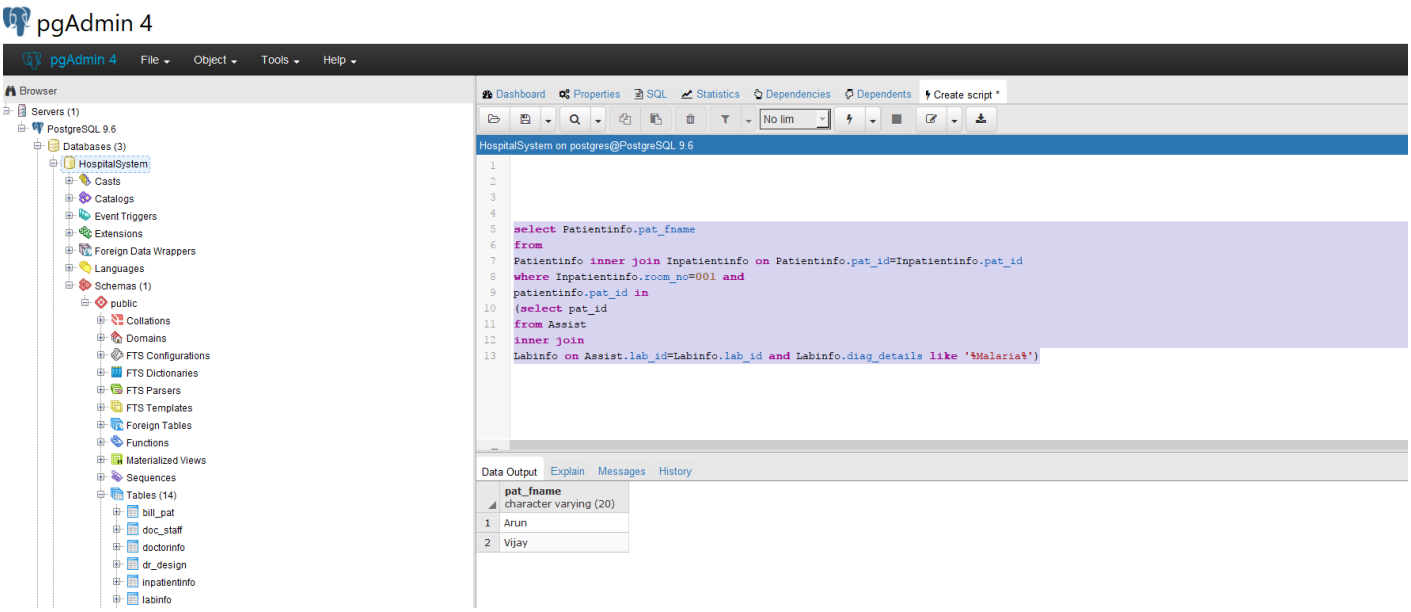
patientinfo.pat_id in

(select pat_id

from Assist

inner join

Labinfo on Assist.lab_id=Labinfo.lab_id and Labinfo.diag_details like '%Malaria%');



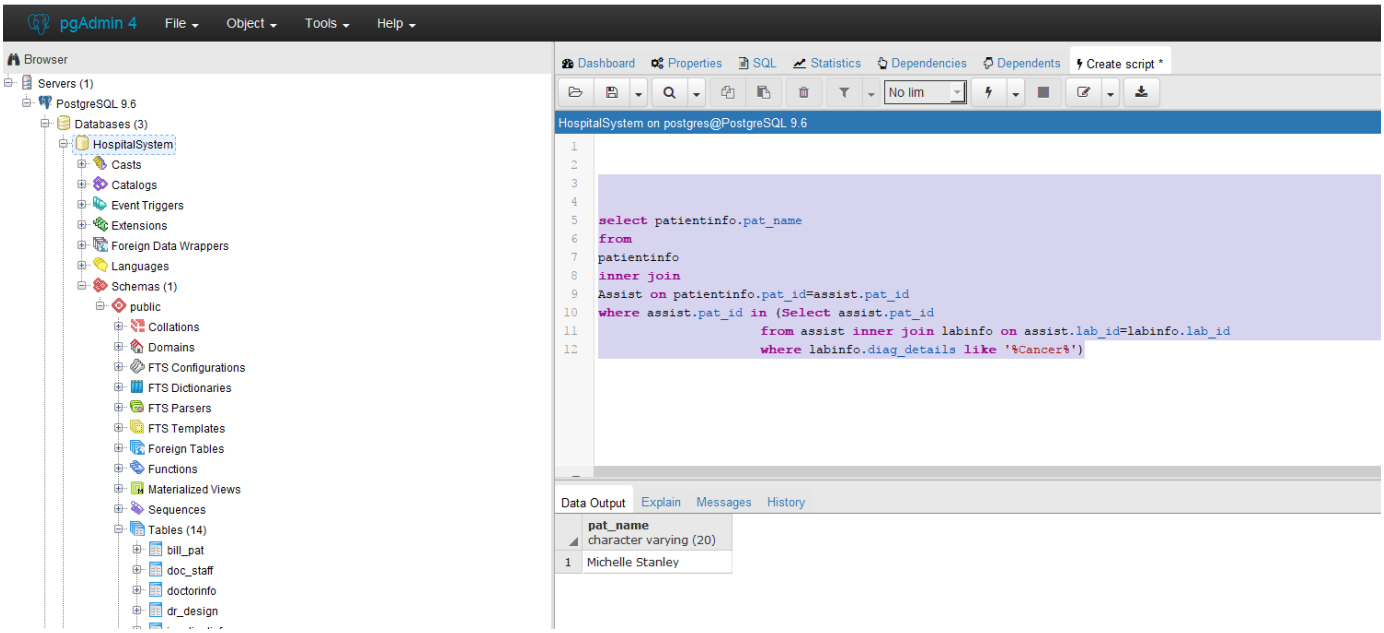
User 2-For Doctor:

Query 1

1) List the name of the patients that have symptoms for cancer?

```
select patientinfo.pat_name
from
patientinfo
inner join
Assist on patientinfo.pat_id=assist.pat_id
where assist.pat_id in (Select assist.pat_id
        from assist inner join labinfo on assist.lab_id=labinfo.lab_id
        where labinfo.diag_details like '%Cancer%')
```

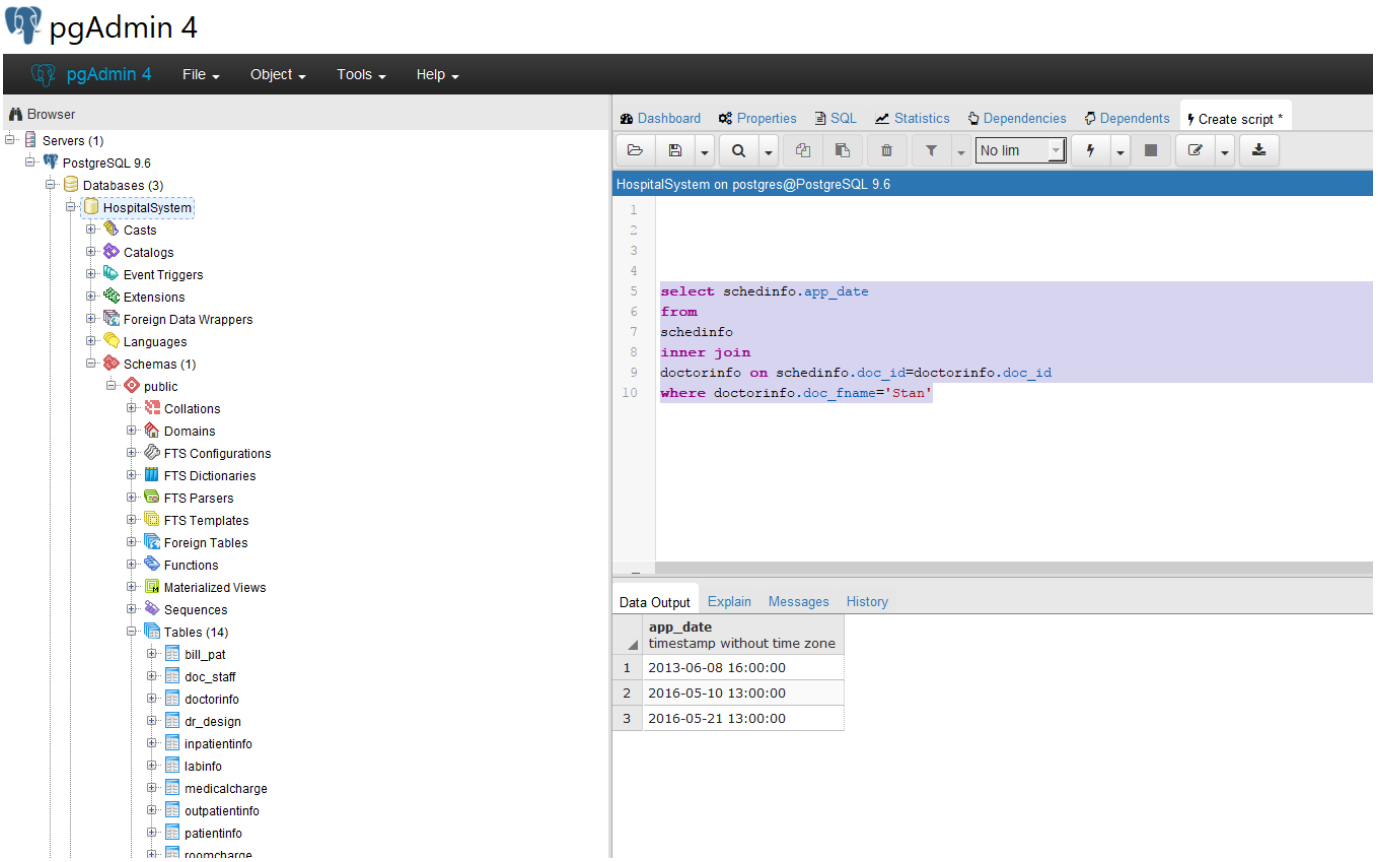
pgAdmin 4



Query 2

2) List the appointments that had been scheduled so far with doctor Stan?

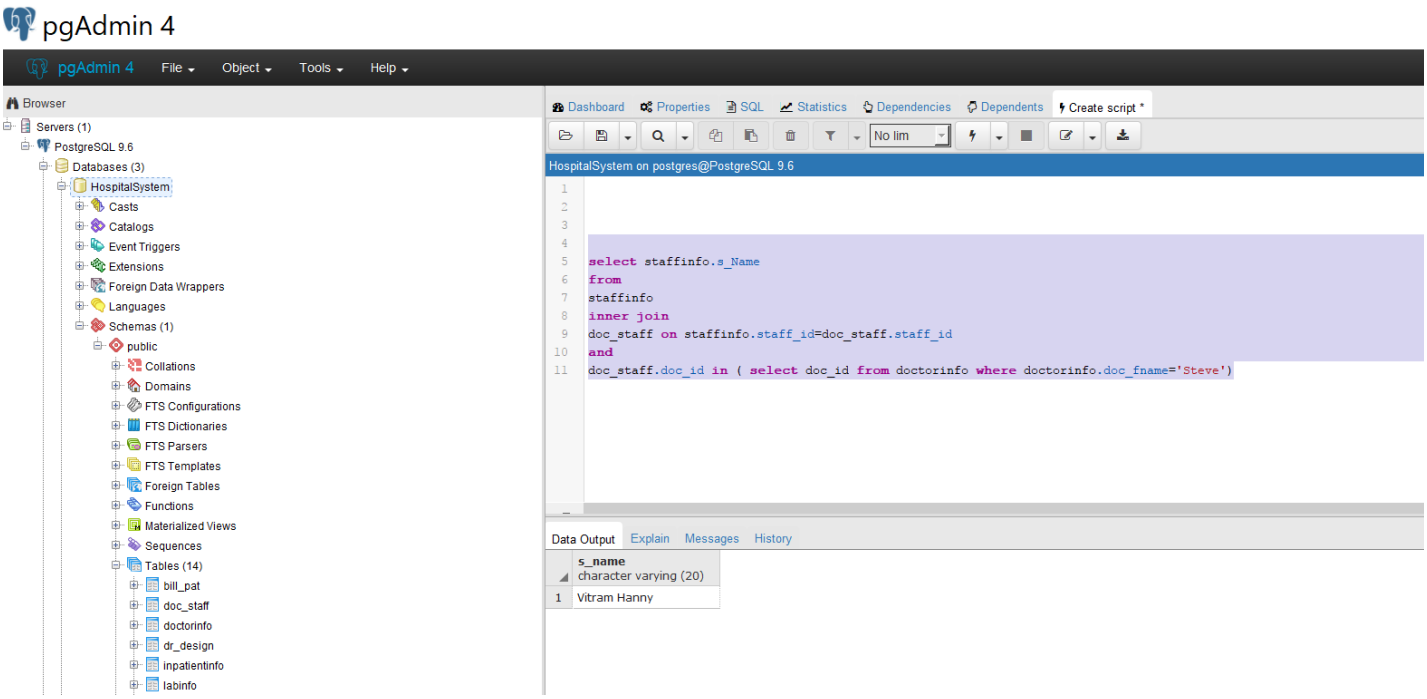
```
select schedinfo.app_date
from
schedinfo
inner join
doctorinfo on schedinfo.doc_id=doctorinfo.doc_id
where doctorinfo.doc_fname='Stan'
```



Query 3

3) List the name of the staff that has been working with Dr Steve?

```
select staffinfo.s_Name
from
staffinfo
inner join
doc_staff on staffinfo.staff_id=doc_staff.staff_id
and
doc_staff.doc_id in ( select doc_id from doctorinfo where doctorinfo.doc_fname='Steve')
```



User 3-For Accountant:

Query 1

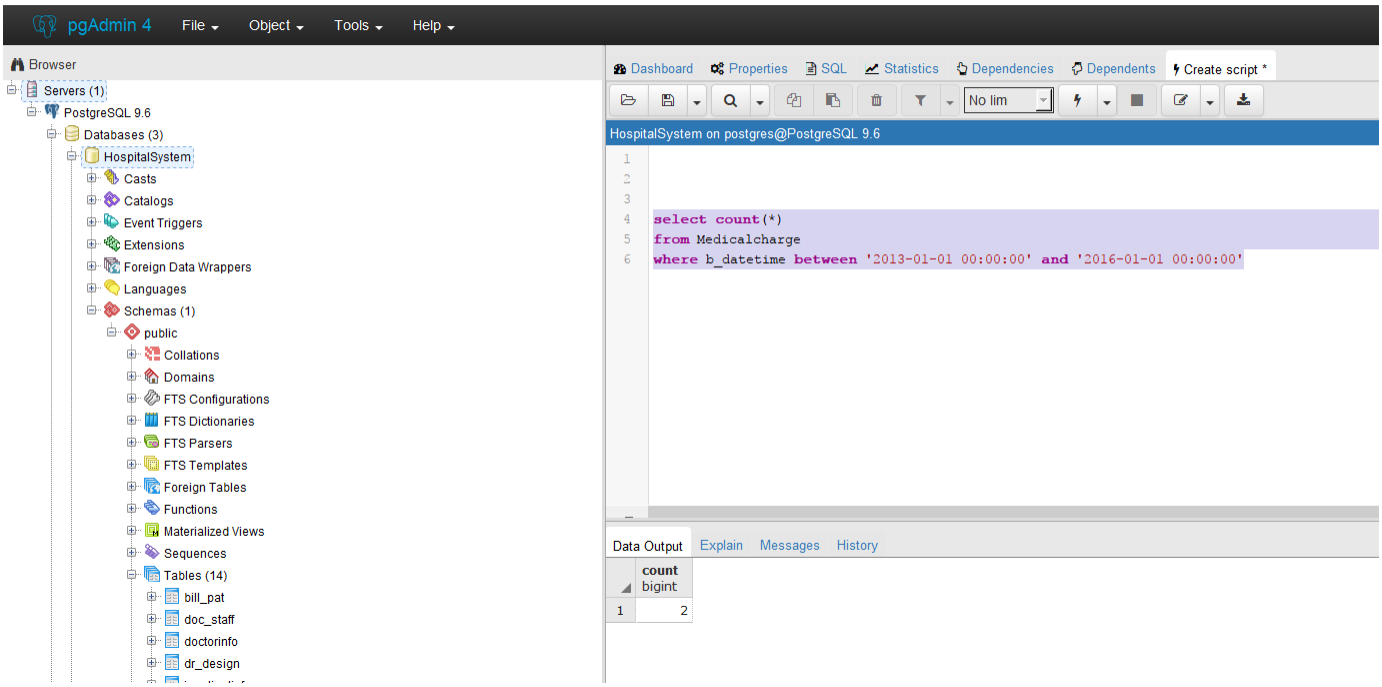
1) List the number of patients whose billing have been initiated between the year 2013 and 2016?

```
select count(*)

from Medicalcharge

where b_datetime between '2013-01-01 00:00:00' and '2016-01-01 00:00:00'
```

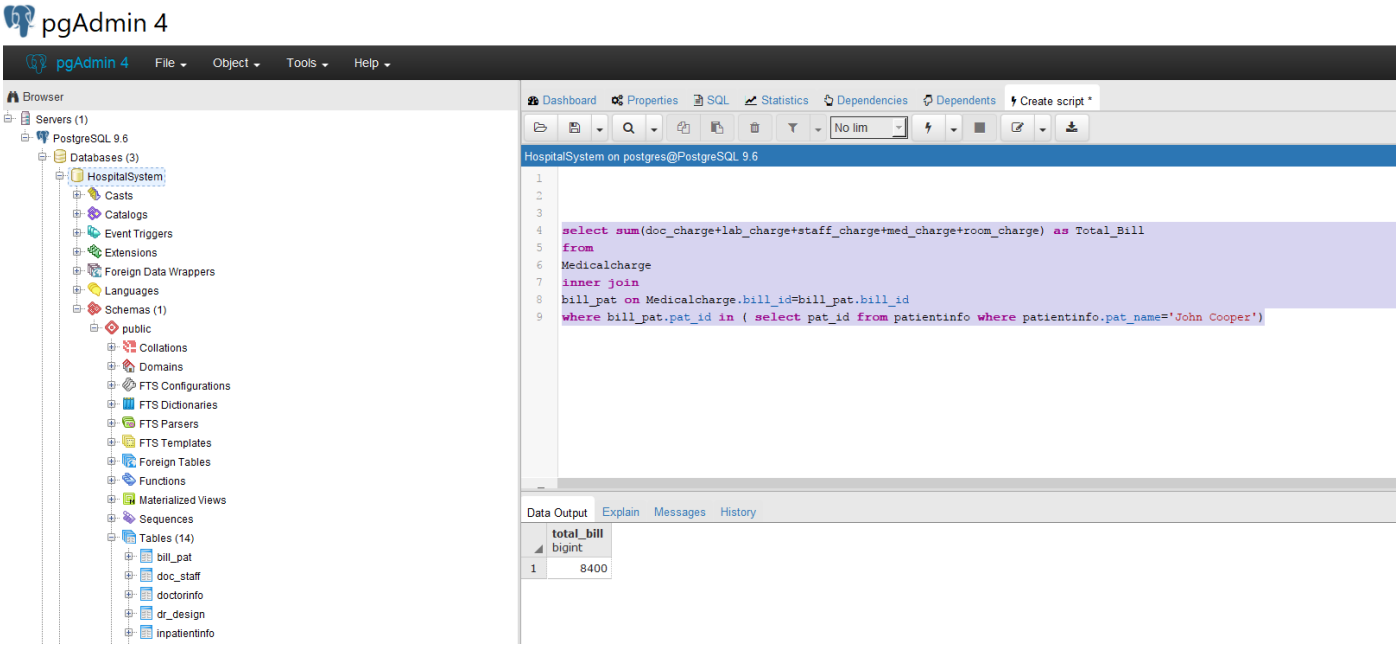
pgAdmin 4



Query 2

2) List the total bill of the patient with name John Cooper?

```
select sum(doc_charge+lab_charge+staff_charge+med_charge+room_charge) as Total_Bill
from
Medicalcharge
inner join
bill_pat on Medicalcharge.bill_id=bill_pat.bill_id
where bill_pat.pat_id in ( select pat_id from patientinfo where patientinfo.pat_name='John Cooper')
```



Query 3

3) Find out the number of bills for the patient with name Arun?

select count(*)

from

Medicalcharge

inner join bill_pat on medicalcharge.bill_id=bill_pat.bill_id where bill_pat.pat_id in (select pat_id from patientinfo where pat_fname='Arun')