Note: We have worked on the HospitalDB\_New database with 17 tables. After creating triggers some of our tables got changed. But to solve other queries, we have restored the original database again.

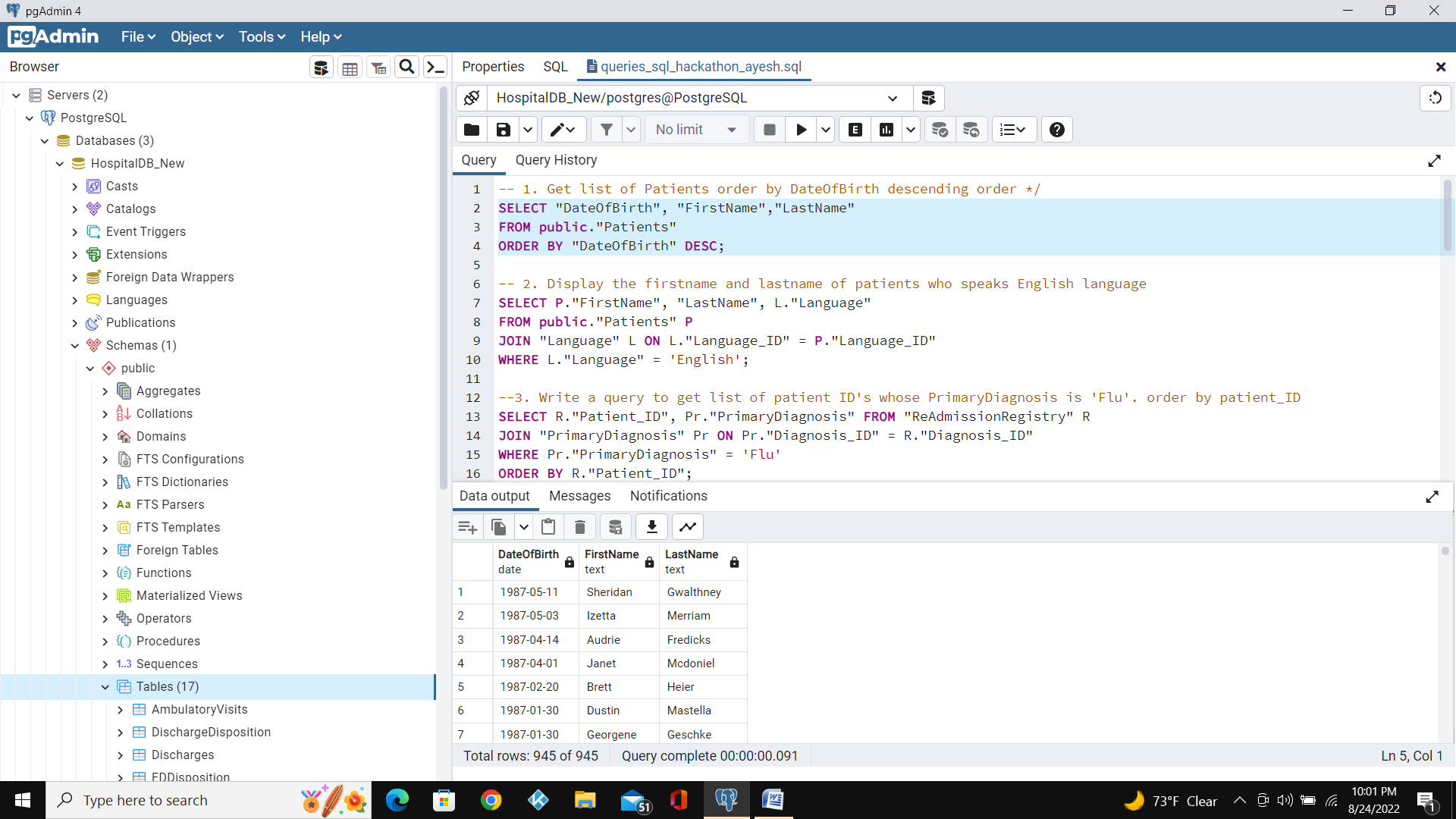
1.Get list of Patients order by DateOfBirth descending order.

Query:

SELECT "DateOfBirth", "FirstName","LastName"

FROM public."Patients"

ORDER BY "DateOfBirth" DESC;



2. Display the firstname and lastname of patients who speaks English language.

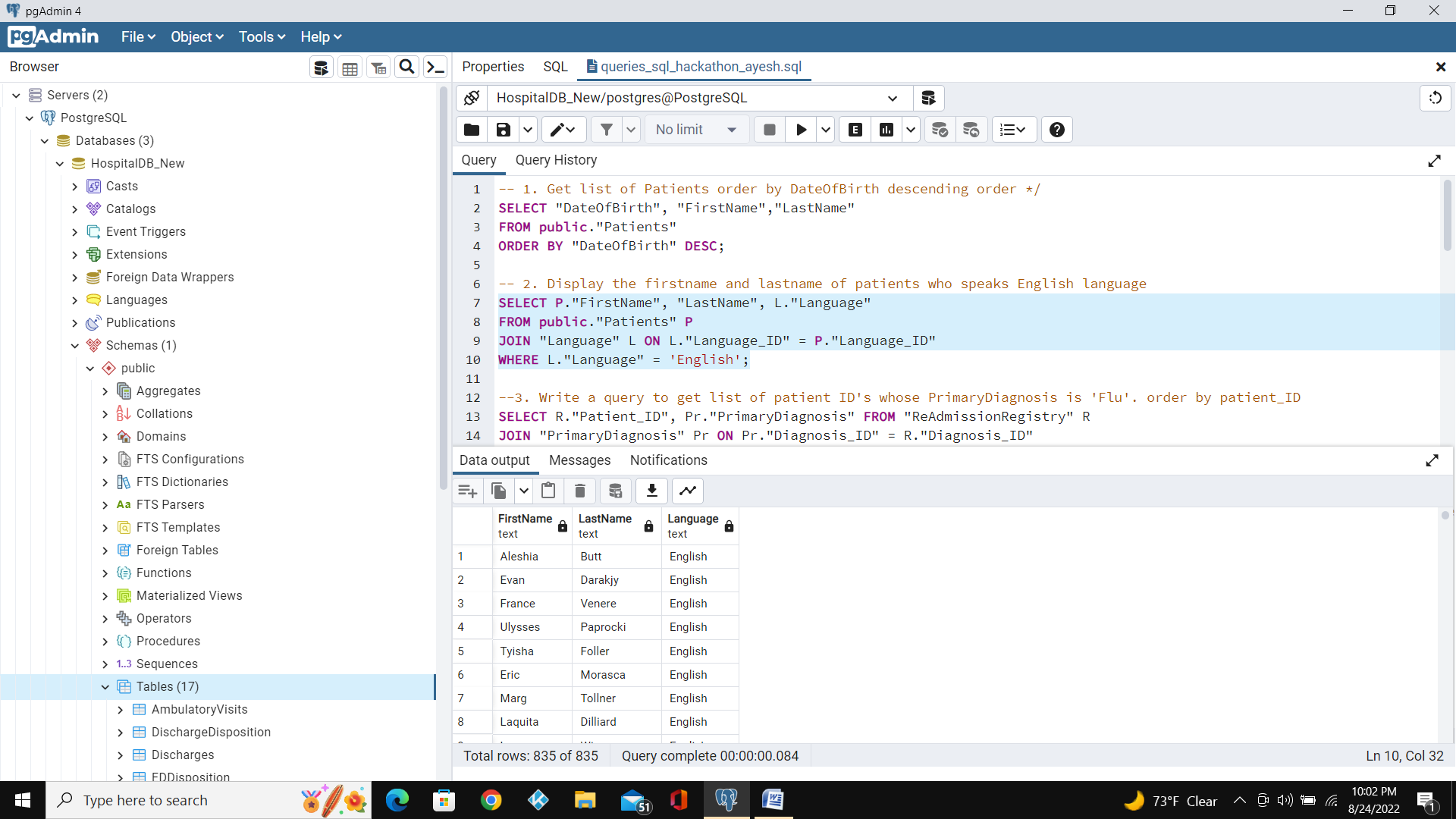
Query:

SELECT P."FirstName",P."LastName", L."Language"

FROM public."Patients" P

Left JOIN "Language" L ON L."Language\_ID" = P."Language\_ID"

WHERE L."Language" = 'English';



3. Write a query to get list of patient ID's whose PrimaryDiagnosis is 'Flu'. order by patient\_ID.

Query:

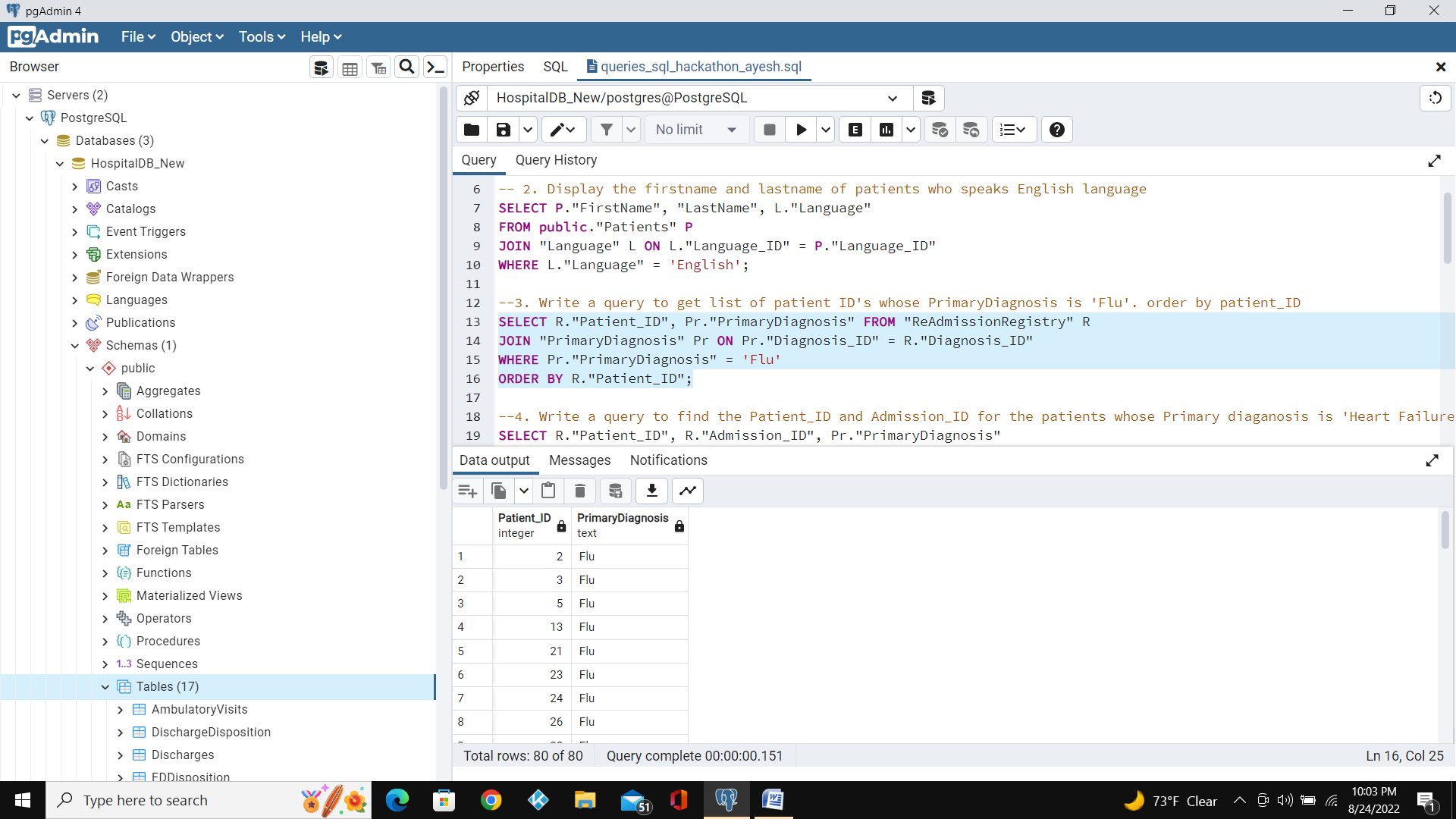
SELECT R."Patient\_ID", Pr."PrimaryDiagnosis"

FROM "ReAdmissionRegistry" R

left JOIN "PrimaryDiagnosis" Pr ON Pr."Diagnosis\_ID" = R."Diagnosis\_ID"

WHERE Pr."PrimaryDiagnosis" = 'Flu'

ORDER BY R."Patient\_ID";



4.Write a query to find the Patient\_ID and Admission\_ID for the patients whose Primary diaganosis is 'Heart Failure'.

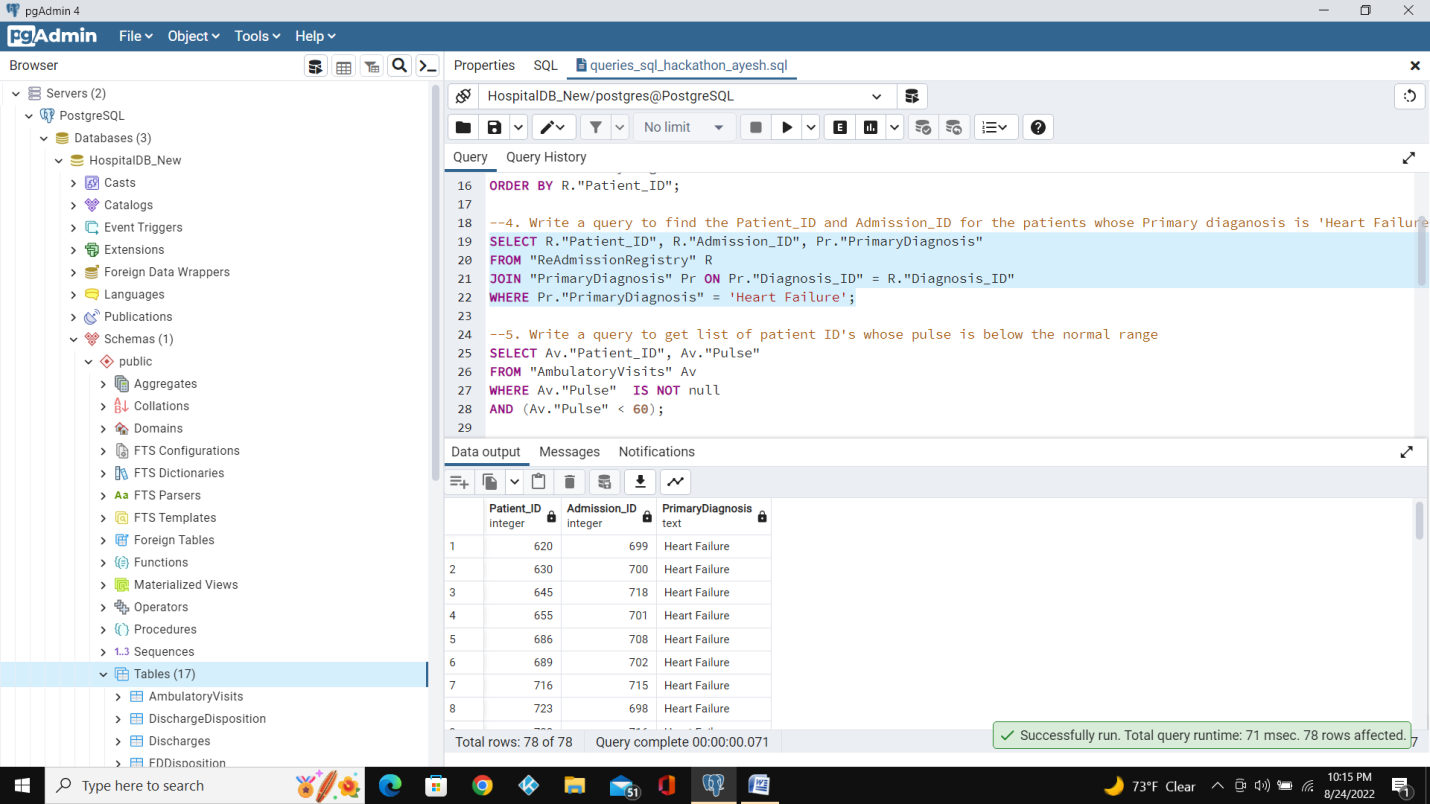
Query:

SELECT R."Patient\_ID", R."Admission\_ID", Pr."PrimaryDiagnosis"

FROM "ReAdmissionRegistry" R

Left JOIN "PrimaryDiagnosis" Pr ON Pr."Diagnosis\_ID" = R."Diagnosis\_ID"

WHERE Pr."PrimaryDiagnosis" = 'Heart Failure';



5. Write a query to get list of patient ID's whose pulse is below the normal range.

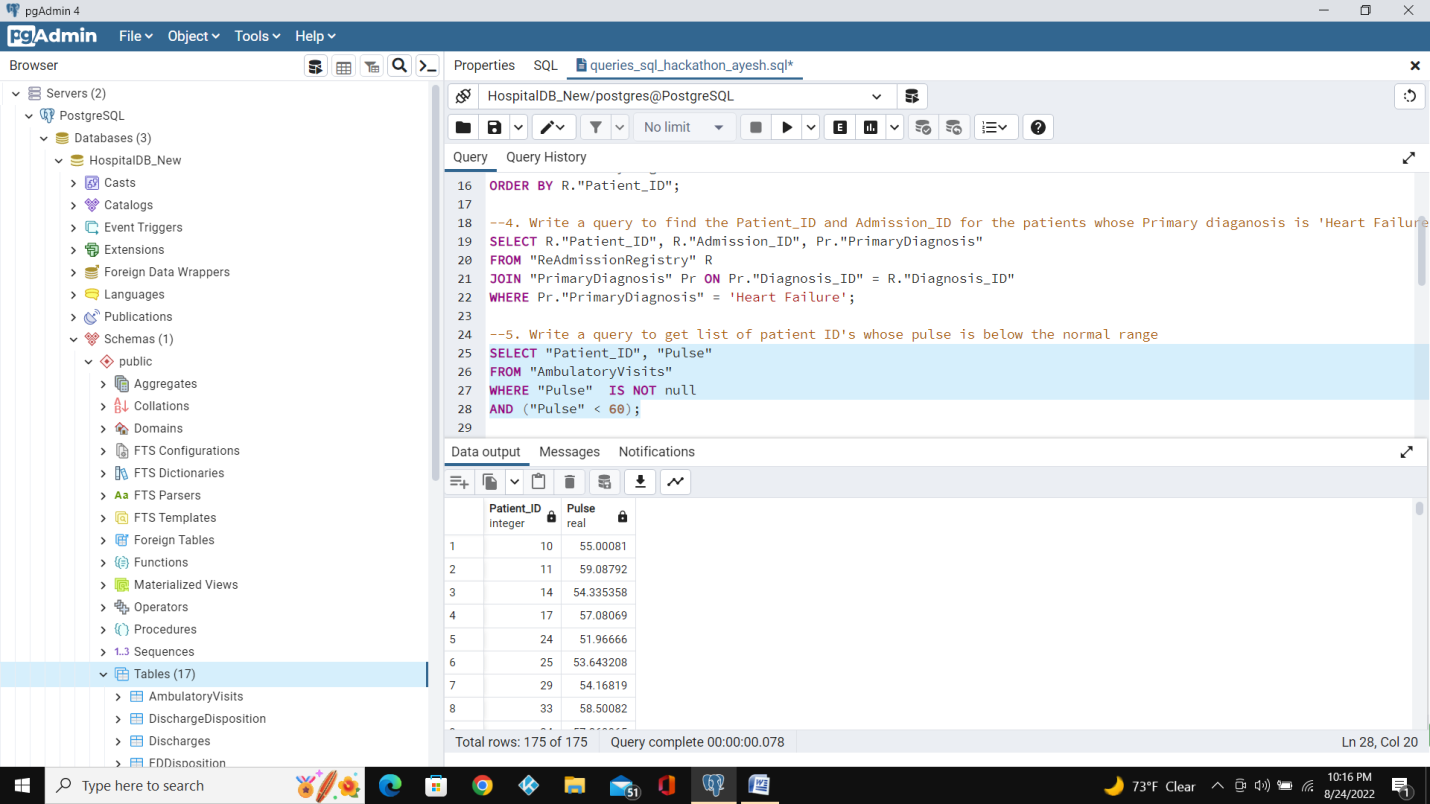
Query:

SELECT "Patient\_ID", "Pulse"

FROM "AmbulatoryVisits"

WHERE "Pulse" IS NOT null

AND ("Pulse" < 60);



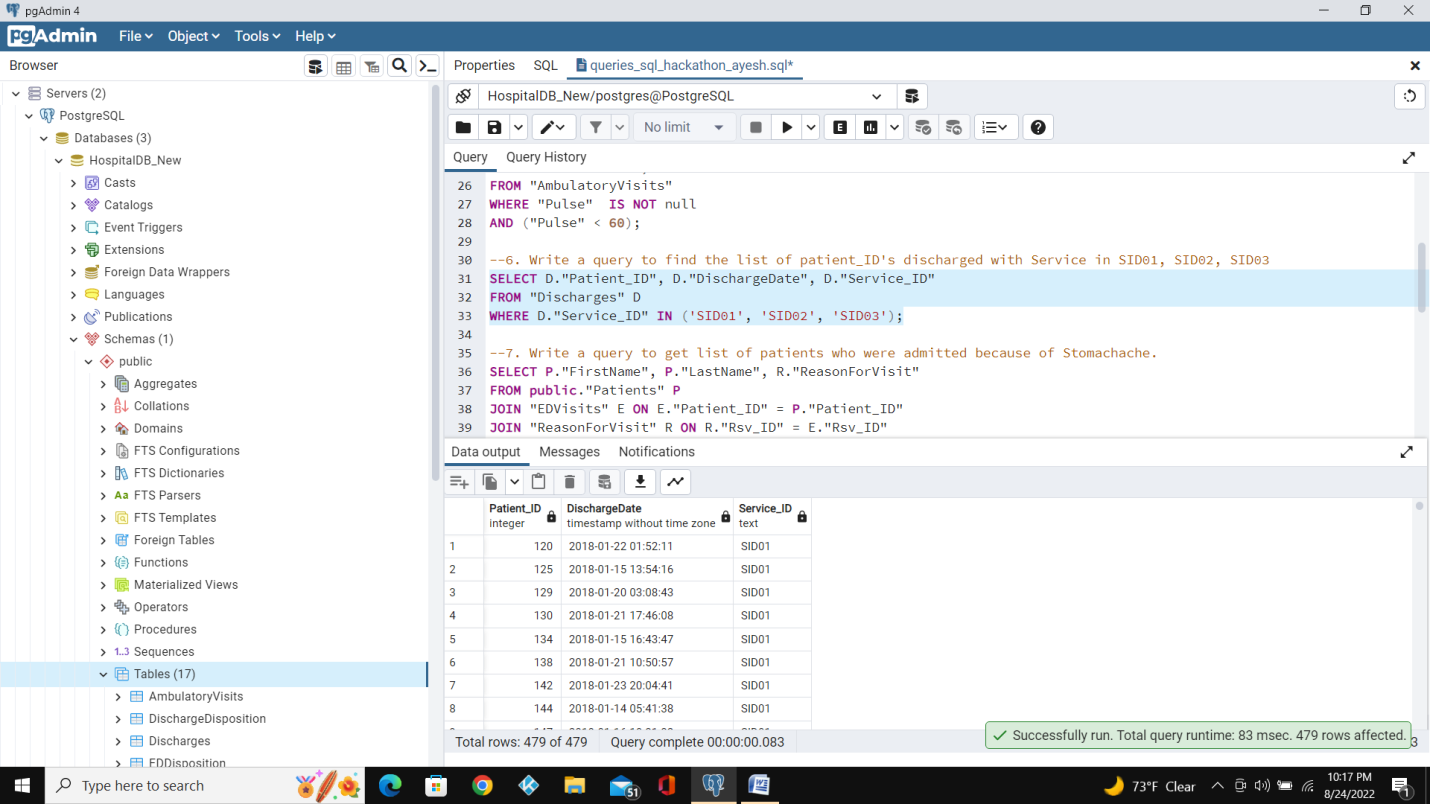
6. Write a query to find the list of patient\_ID's discharged with Service in SID01, SID02, SID03.

Query:

SELECT D."Patient\_ID", D."DischargeDate", D."Service\_ID"

FROM "Discharges" D

WHERE D."Service\_ID" IN ('SID01', 'SID02', 'SID03');



7. Write a query to get list of patients who were admitted because of Stomachache.

Query:

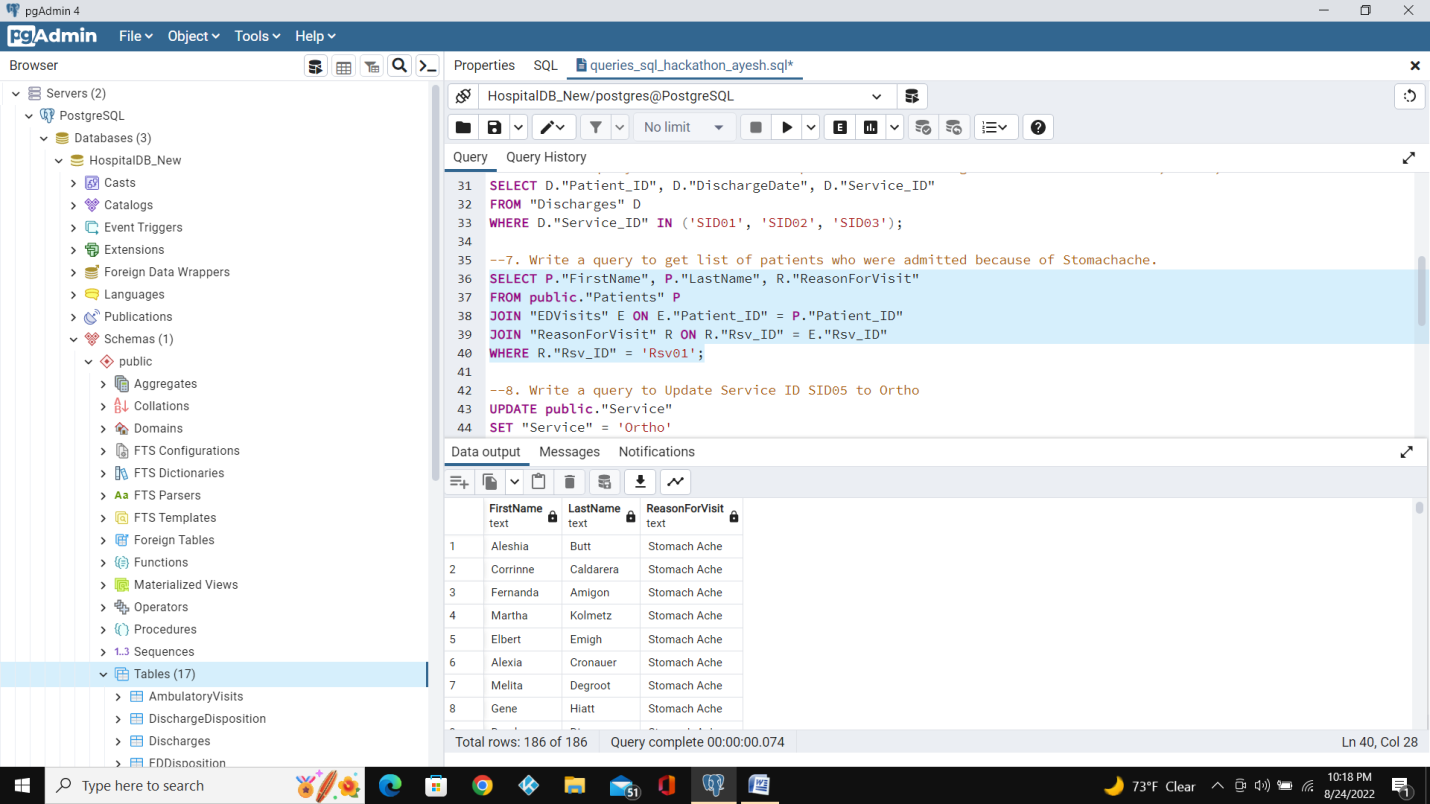
SELECT P."FirstName", P."LastName", R."ReasonForVisit"

FROM public."Patients" P

Left JOIN "EDVisits" E ON E."Patient\_ID" = P."Patient\_ID"

Left JOIN "ReasonForVisit" R ON R."Rsv\_ID" = E."Rsv\_ID"

WHERE R."Rsv\_ID" = 'Rsv01';



8. Write a query to Update Service ID SID05 to Ortho.

Query:

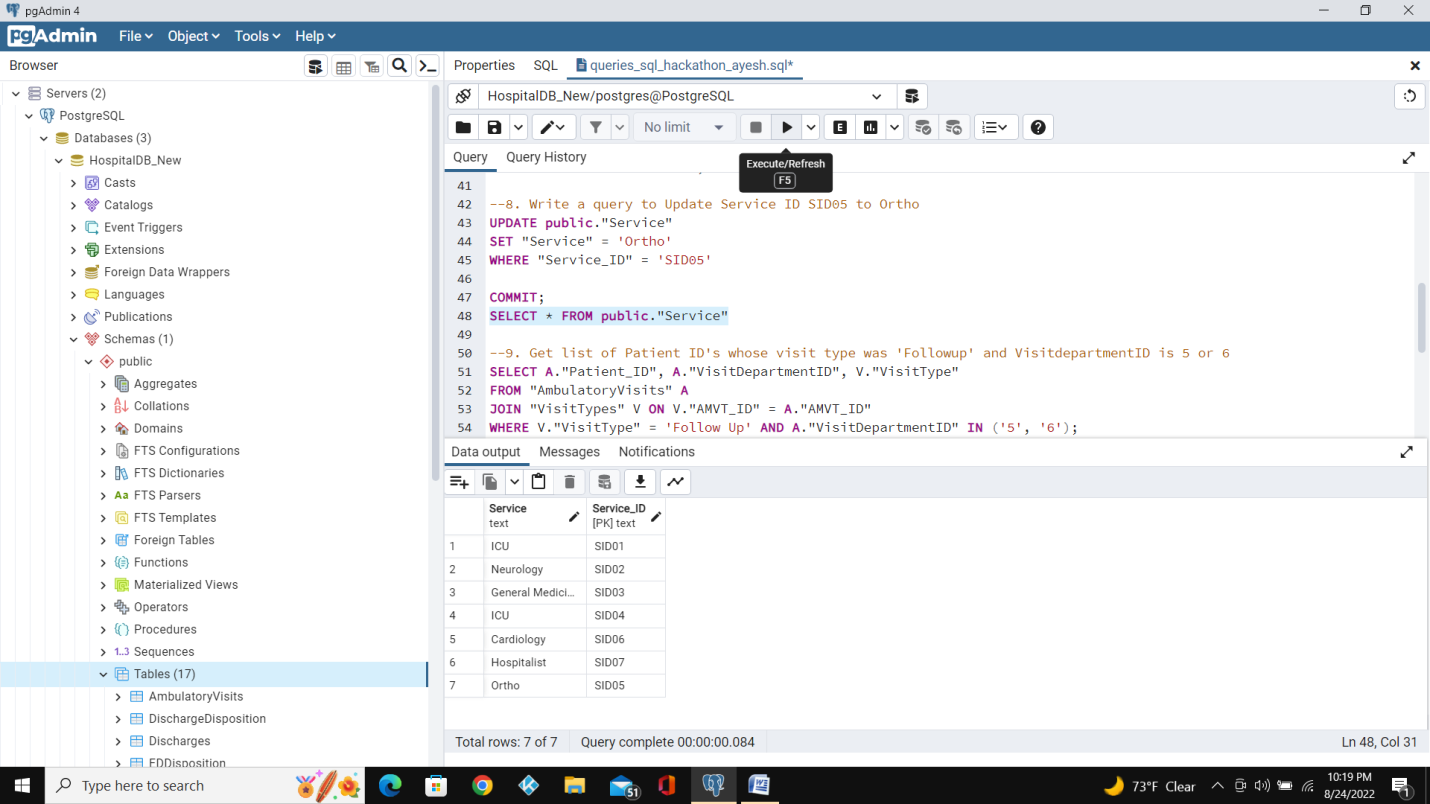
UPDATE public."Service"

SET "Service" = 'Ortho'

WHERE "Service\_ID" = 'SID05'

COMMIT;

SELECT \* FROM public."Service";



9. Get list of Patient ID's whose visit type was 'Followup' and VisitdepartmentID is 5 or 6

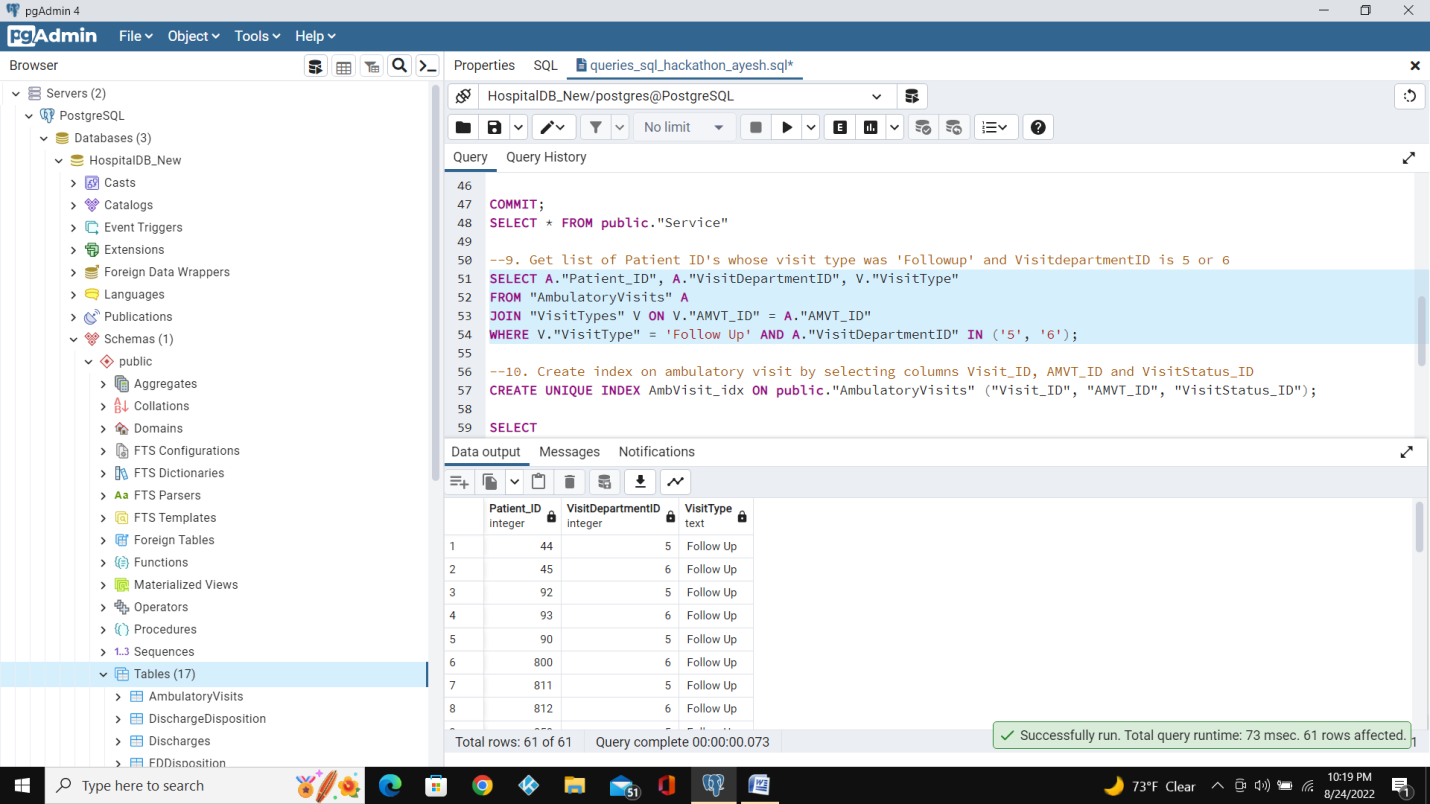
Query:

SELECT A."Patient\_ID", A."VisitDepartmentID", V."VisitType"

FROM "AmbulatoryVisits" A

LEFT JOIN "VisitTypes" V ON V."AMVT\_ID" = A."AMVT\_ID"

WHERE V."VisitType" = 'Follow Up' AND A."VisitDepartmentID" IN ('5', '6');



10. Create index on ambulatory visit by selecting columns Visit\_ID, AMVT\_ID and VisitStatus\_ID.

Query:

CREATE UNIQUE INDEX AmbVisit\_idx ON public."AmbulatoryVisits" ("Visit\_ID", "AMVT\_ID", "VisitStatus\_ID");

SELECT

indexname,

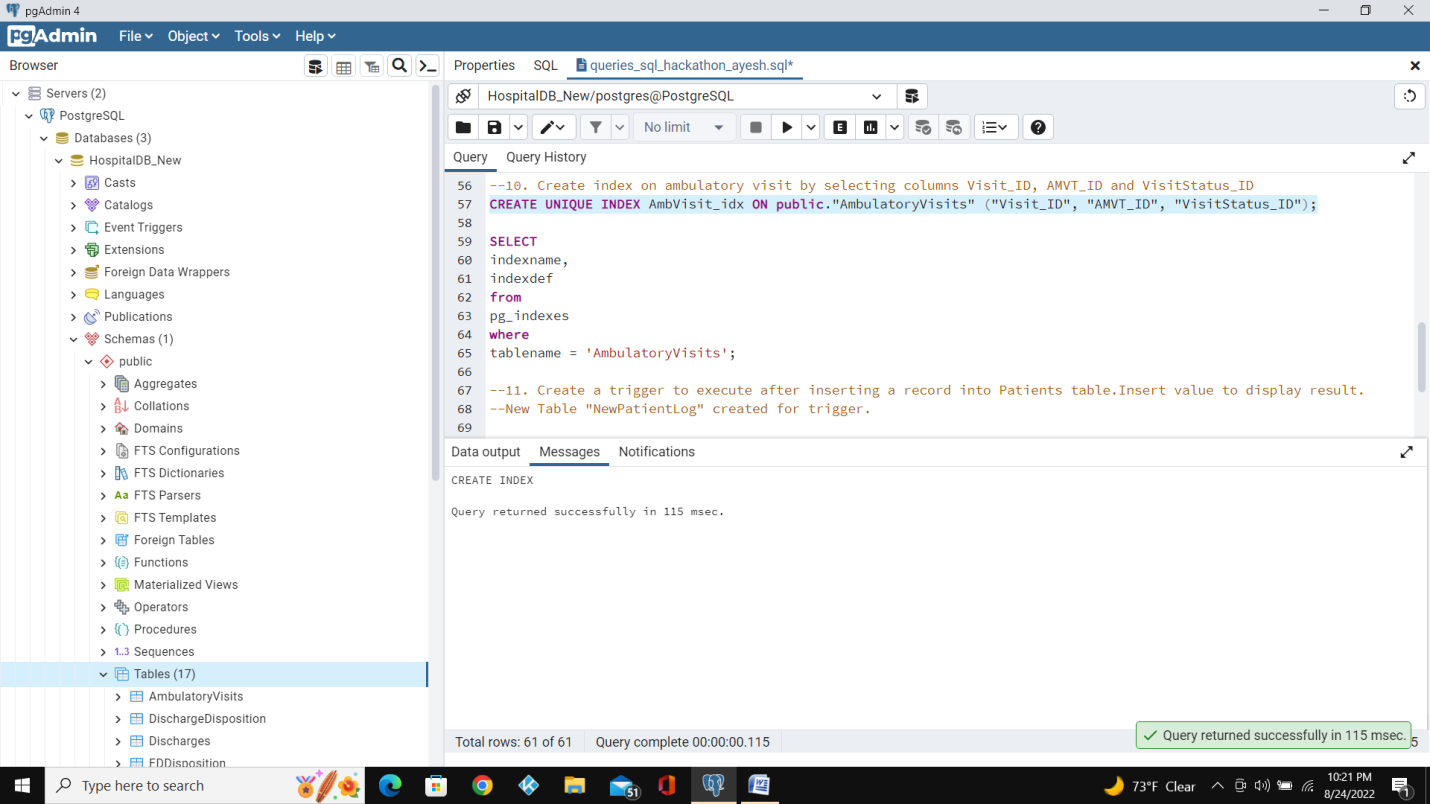
indexdef

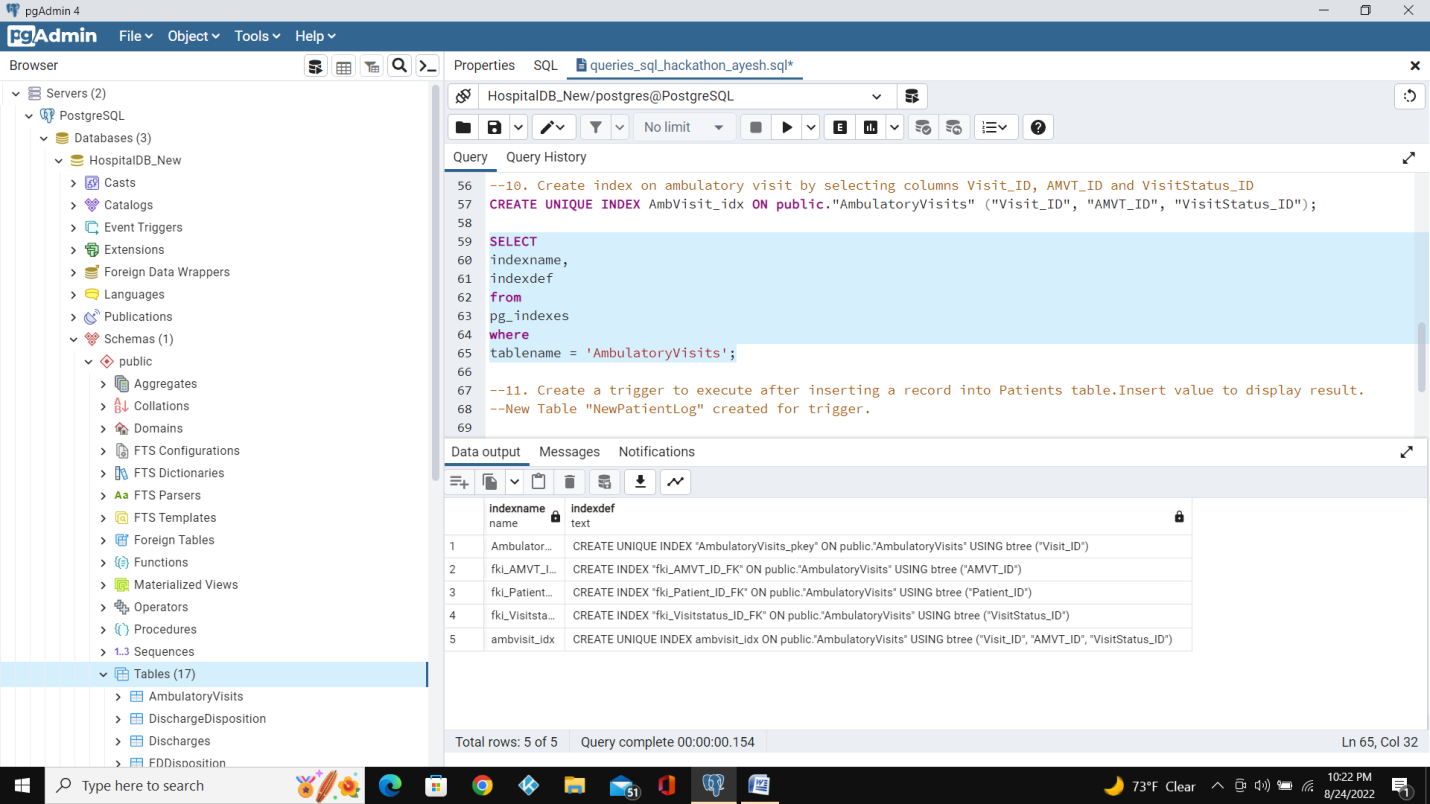
from

pg\_indexes

where

tablename = 'AmbulatoryVisits';





11. Create a trigger to execute after inserting a record into Patients table.Insert value to display result.

Explanation:New Table "NewPatientLog" created for trigger.

Query:

CREATE TABLE IF NOT EXISTS public."NewPatientLog"

(

"Patient\_ID" integer NOT NULL,

"FirstName" text COLLATE pg\_catalog."default",

"LastName" text COLLATE pg\_catalog."default",

"DateOfBirth" date,

"Gender\_ID" text COLLATE pg\_catalog."default",

"Race\_ID" text COLLATE pg\_catalog."default",

"Language\_ID" text COLLATE pg\_catalog."default",

CONSTRAINT "NewPatientLog\_pkey" PRIMARY KEY ("Patient\_ID"),

CONSTRAINT "NewPatientLogGender\_ID\_FK" FOREIGN KEY ("Gender\_ID")

REFERENCES public."Gender" ("Gender\_ID") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "NewPatientLogLangugae\_ID\_FK" FOREIGN KEY ("Language\_ID")

REFERENCES public."Language" ("Language\_ID") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION,

CONSTRAINT "NewPatientLogRace\_ID\_FK" FOREIGN KEY ("Race\_ID")

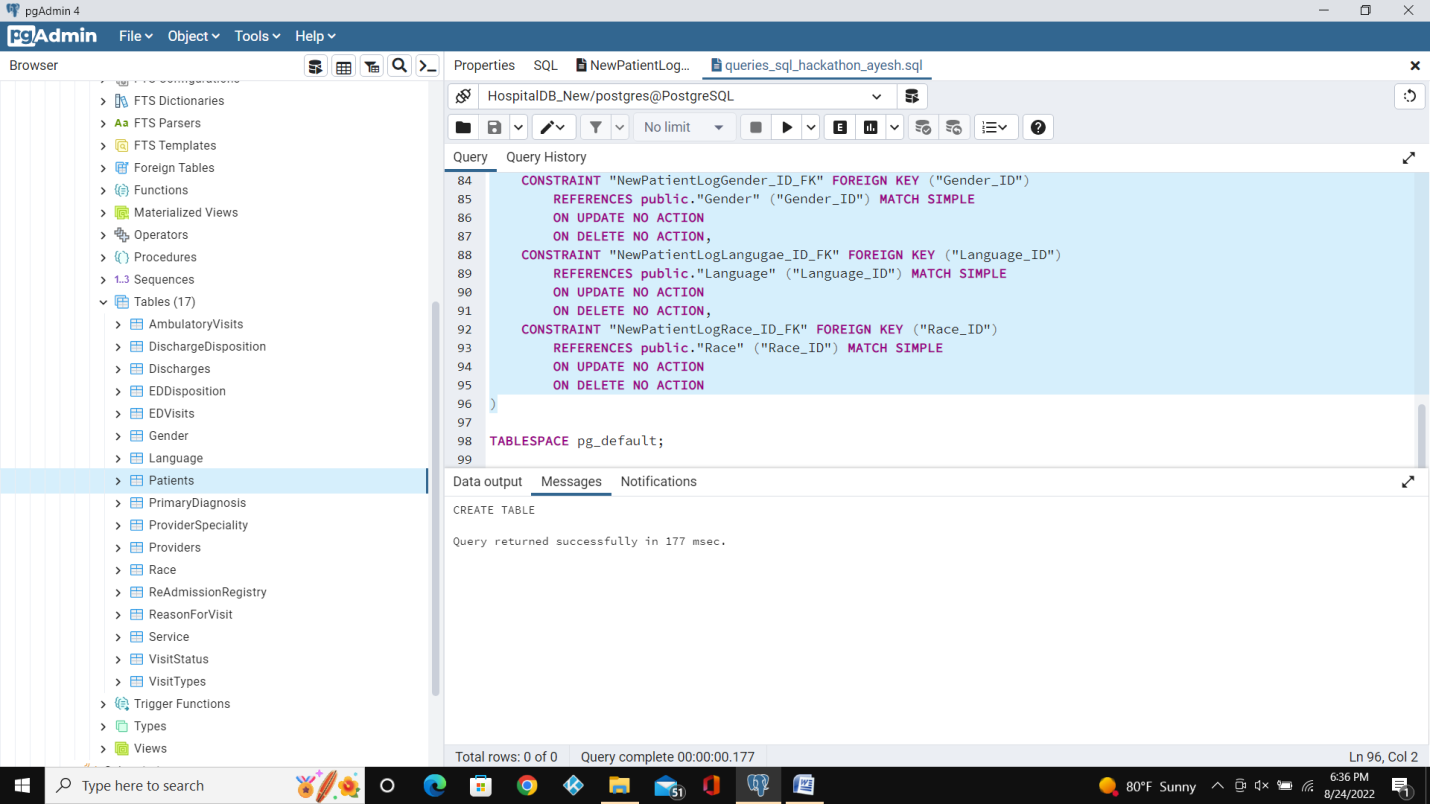
REFERENCES public."Race" ("Race\_ID") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION

)

TABLESPACE pg\_default;



ALTER TABLE IF EXISTS public."NewPatientLog"

OWNER to postgres;

-- Index: fki\_Gender\_ID\_FK

-- DROP INDEX IF EXISTS public."NewPatientLogfki\_Gender\_ID\_FK";

CREATE INDEX IF NOT EXISTS "NewPatientLogfki\_Gender\_ID\_FK"

ON public."NewPatientLog" USING btree

("Gender\_ID" COLLATE pg\_catalog."default" ASC NULLS LAST)

TABLESPACE pg\_default;

-- Index: fki\_Langugae\_ID\_FK

-- DROP INDEX IF EXISTS public."NewPatientLogfki\_Langugae\_ID\_FK";

CREATE INDEX IF NOT EXISTS "NewPatientLogfki\_Langugae\_ID\_FK"

ON public."NewPatientLog" USING btree

("Language\_ID" COLLATE pg\_catalog."default" ASC NULLS LAST)

TABLESPACE pg\_default;

-- Index: fki\_Race\_ID\_FK

-- DROP INDEX IF EXISTS public."NewPatientLogfki\_Race\_ID\_FK";

CREATE INDEX IF NOT EXISTS "NewPatientLogfki\_Race\_ID\_FK"

ON public."NewPatientLog" USING btree

("Race\_ID" COLLATE pg\_catalog."default" ASC NULLS LAST)

TABLESPACE pg\_default;

-- Index: index\_patient\_id

-- DROP INDEX IF EXISTS public.NewPatientLogindex\_patient\_id;

CREATE OR REPLACE FUNCTION record\_insert()

RETURNS TRIGGER AS

$$

BEGIN

INSERT INTO public."NewPatientLog"("Patient\_ID", "FirstName", "LastName", "DateOfBirth", "Gender\_ID", "Race\_ID", "Language\_ID")

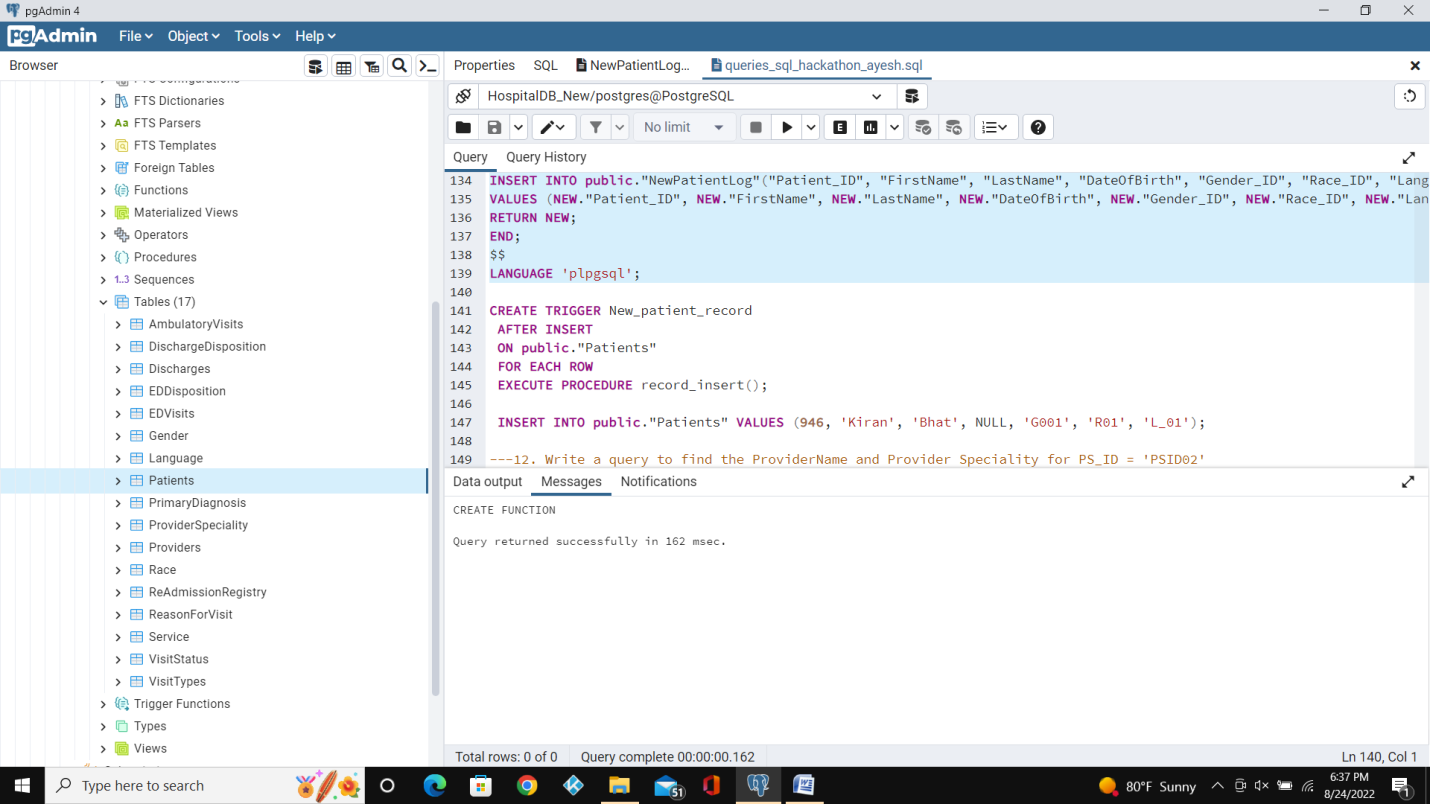
VALUES (NEW."Patient\_ID", NEW."FirstName", NEW."LastName", NEW."DateOfBirth", NEW."Gender\_ID", NEW."Race\_ID", NEW."Language\_ID");

RETURN NEW;

END;

$$

LANGUAGE 'plpgsql';



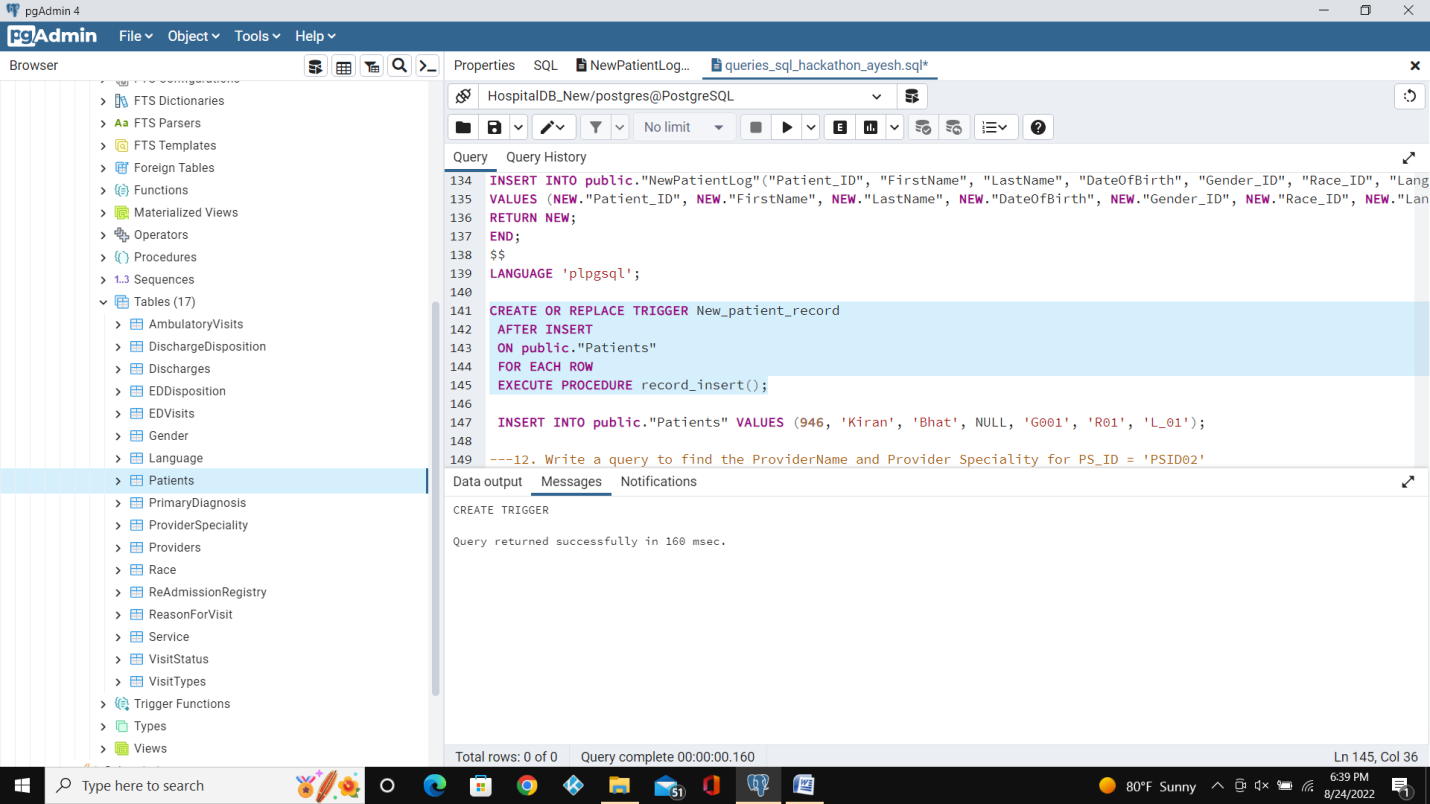
CREATE OR REPLACE TRIGGER New\_patient\_record

AFTER INSERT

ON public."Patients"

FOR EACH ROW

EXECUTE PROCEDURE record\_insert();



Explanation:Inserting values into “Patients” Table.

Query:

INSERT INTO public."Patients" VALUES (946, 'Kiran', 'Bhat', NULL, 'G001', 'R01', 'L\_01');

12. Write a query to find the ProviderName and Provider Speciality for PS\_ID = 'PSID02'.

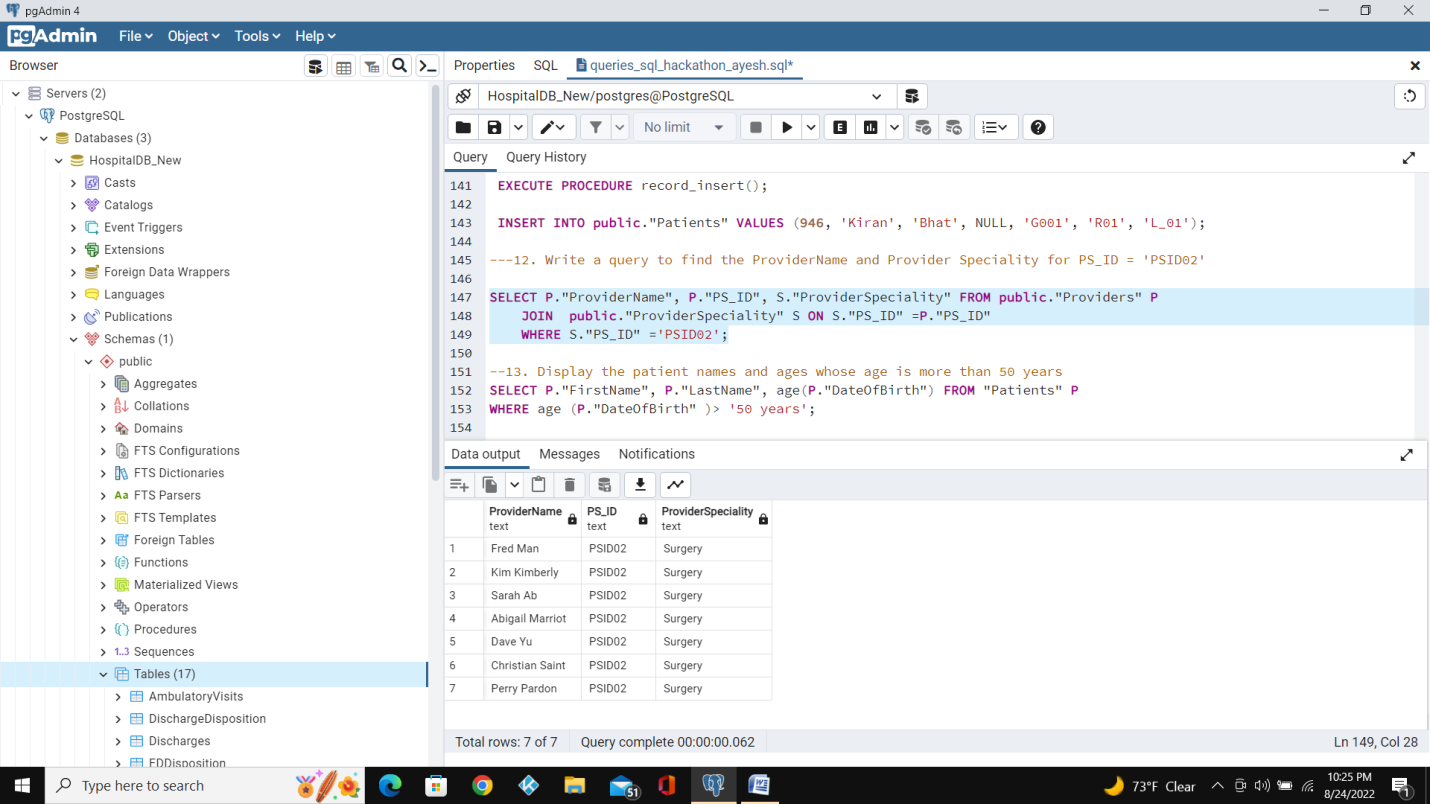
Query:

SELECT P."ProviderName", P."PS\_ID", S."ProviderSpeciality"

FROM public."Providers" P

left JOIN public."ProviderSpeciality" S ON S."PS\_ID" =P."PS\_ID"

WHERE S."PS\_ID" ='PSID02';

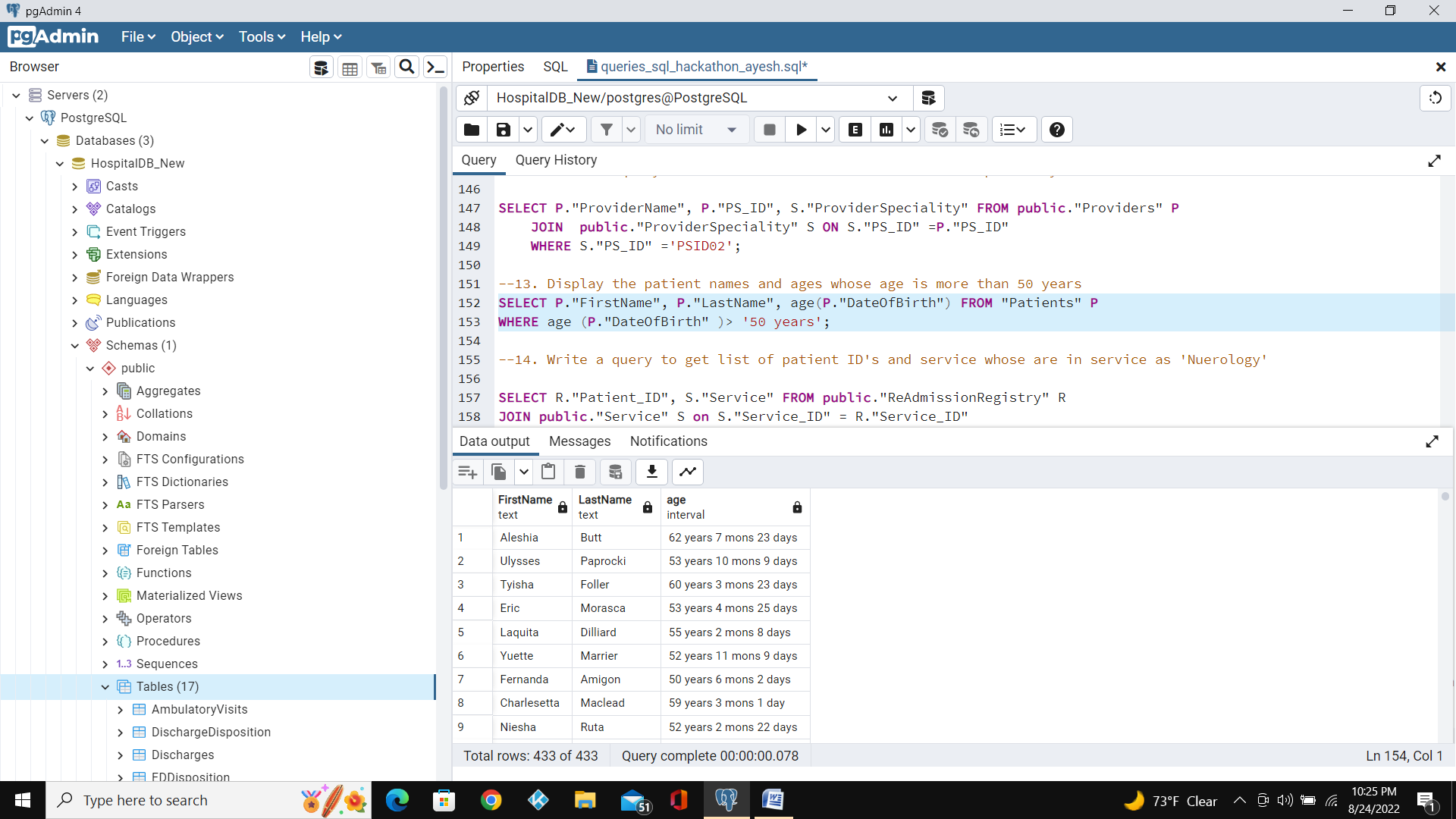


13. Display the patient names and ages whose age is more than 50 years.

Query:

SELECT P."FirstName", P."LastName", age(P."DateOfBirth") FROM "Patients" P

WHERE age (P."DateOfBirth" )> '50 years';



14. Write a query to get list of patient ID'sand service whose are in service as 'Neurology'.

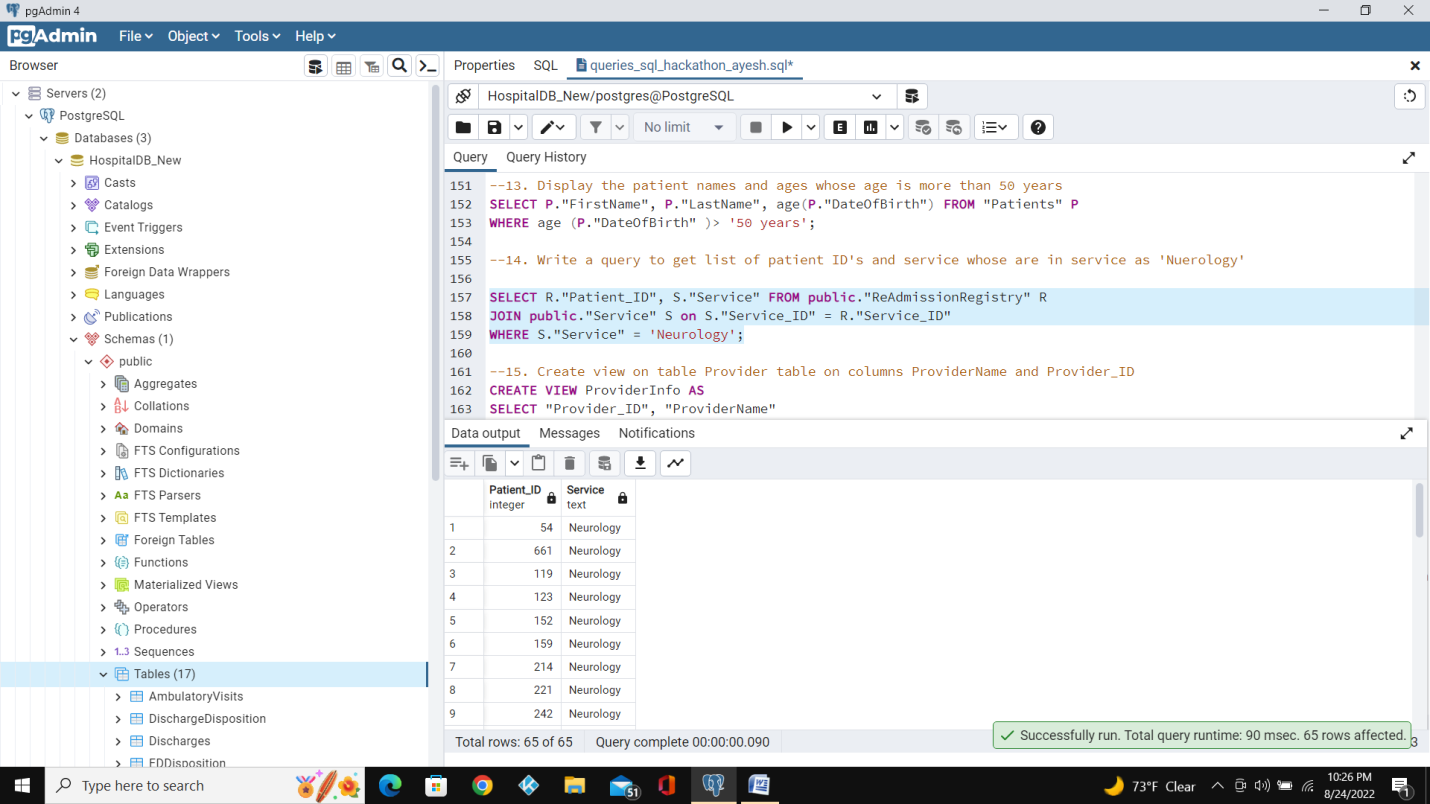
Query:

SELECT R."Patient\_ID", S."Service"

FROM public."ReAdmissionRegistry" R

Left JOIN public."Service" S on S."Service\_ID" = R."Service\_ID"

WHERE S."Service" = 'Neurology';



15. Create view on table Provider table on columns ProviderName and Provider\_ID.

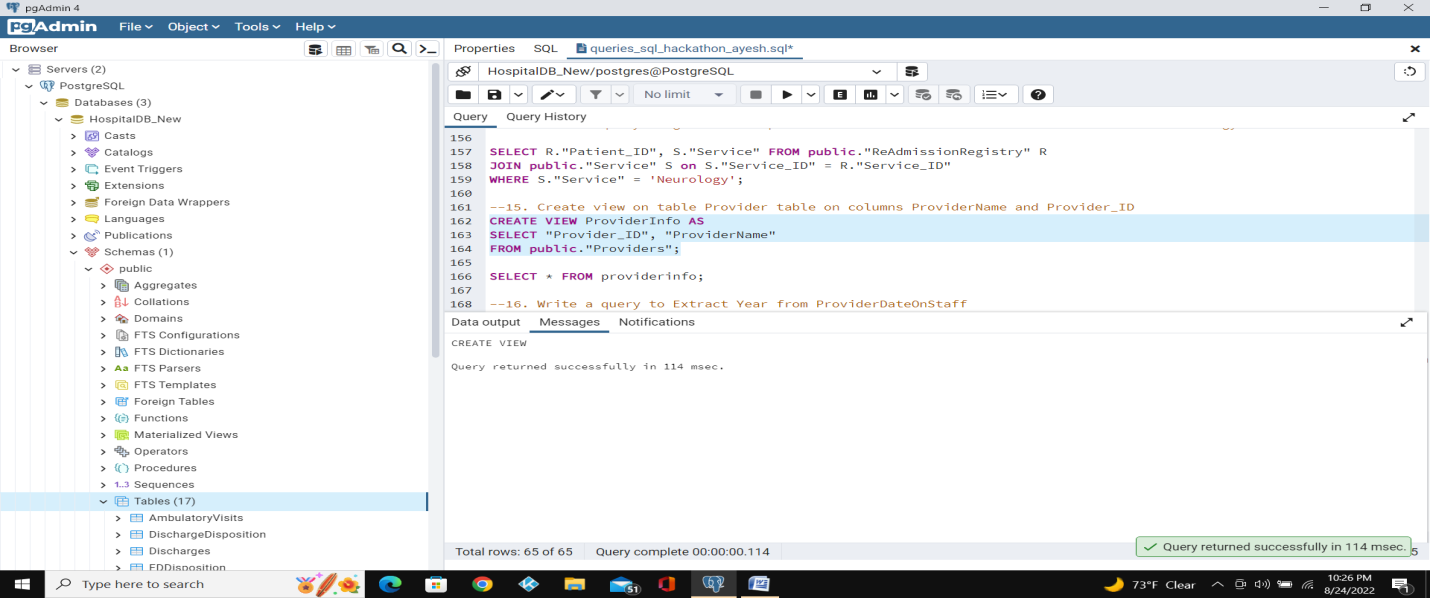
Query:

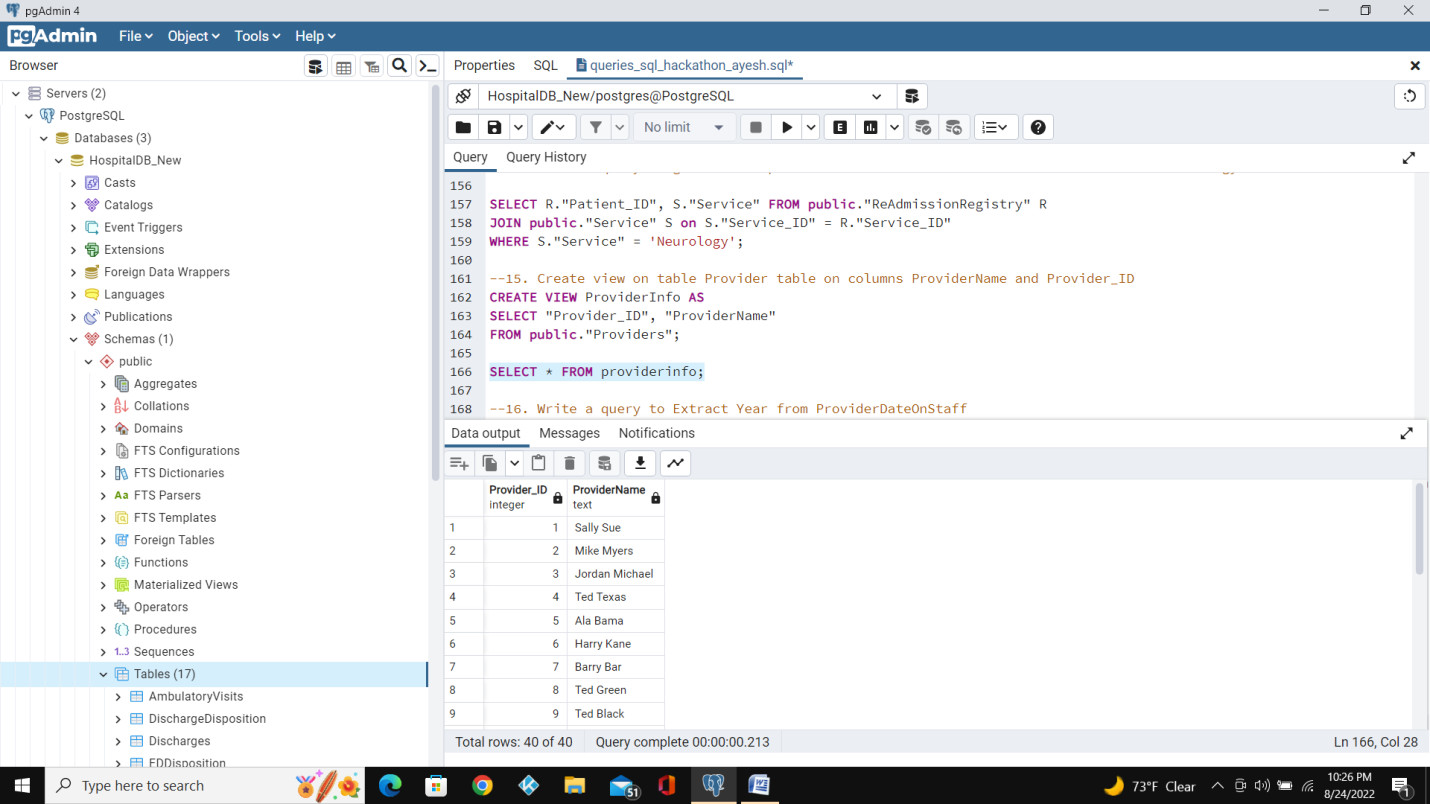
CREATE VIEW ProviderInfo AS

SELECT "Provider\_ID", "ProviderName"

FROM public."Providers";

SELECT \* FROM providerinfo;



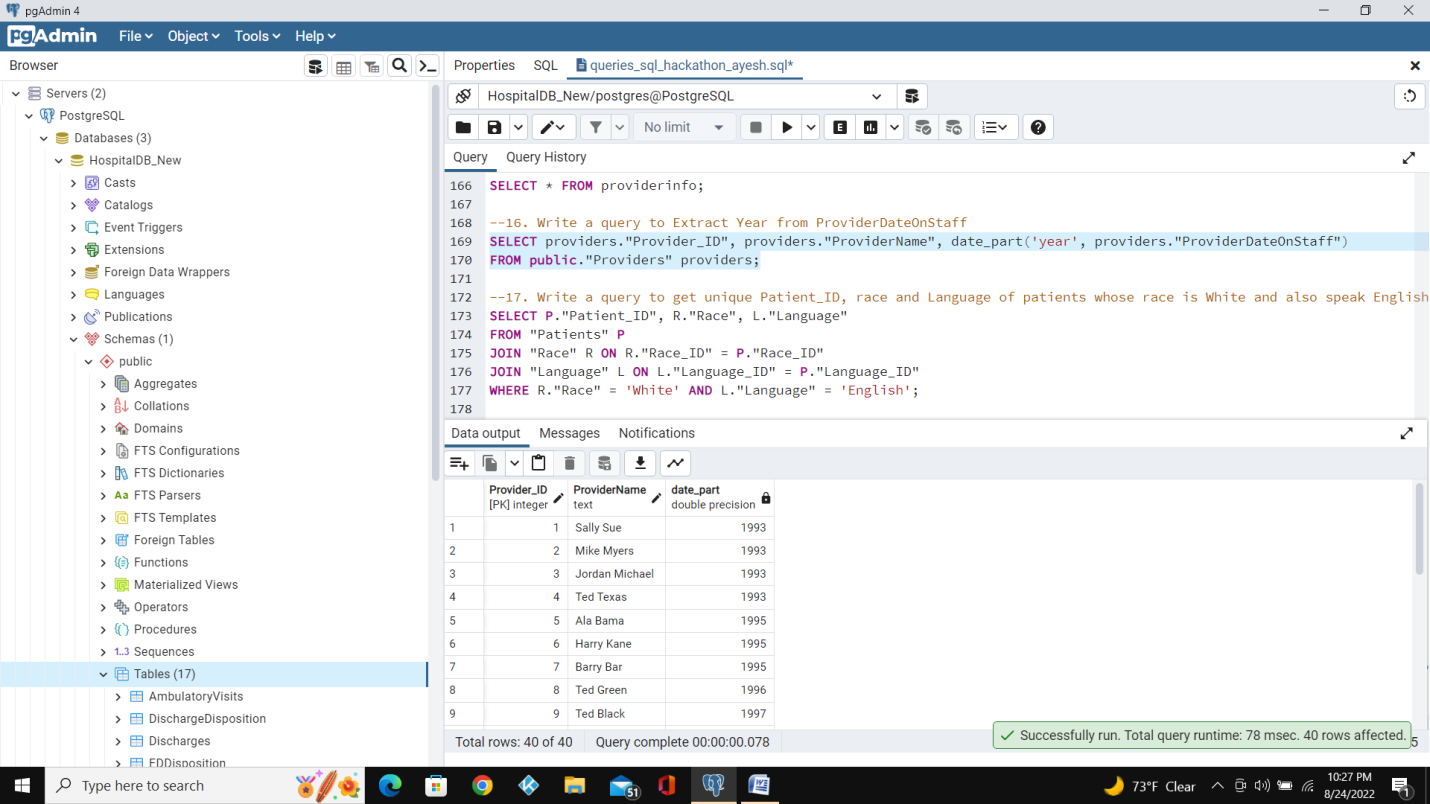


16. Write a query to Extract Year from ProviderDateOnStaff.

Query:

SELECT providers."Provider\_ID", providers."ProviderName", date\_part('year', providers."ProviderDateOnStaff")

FROM public."Providers" providers;



17. Write a query to get unique Patient\_ID,race and Language of patients whose race is White and also speak English.

Query:

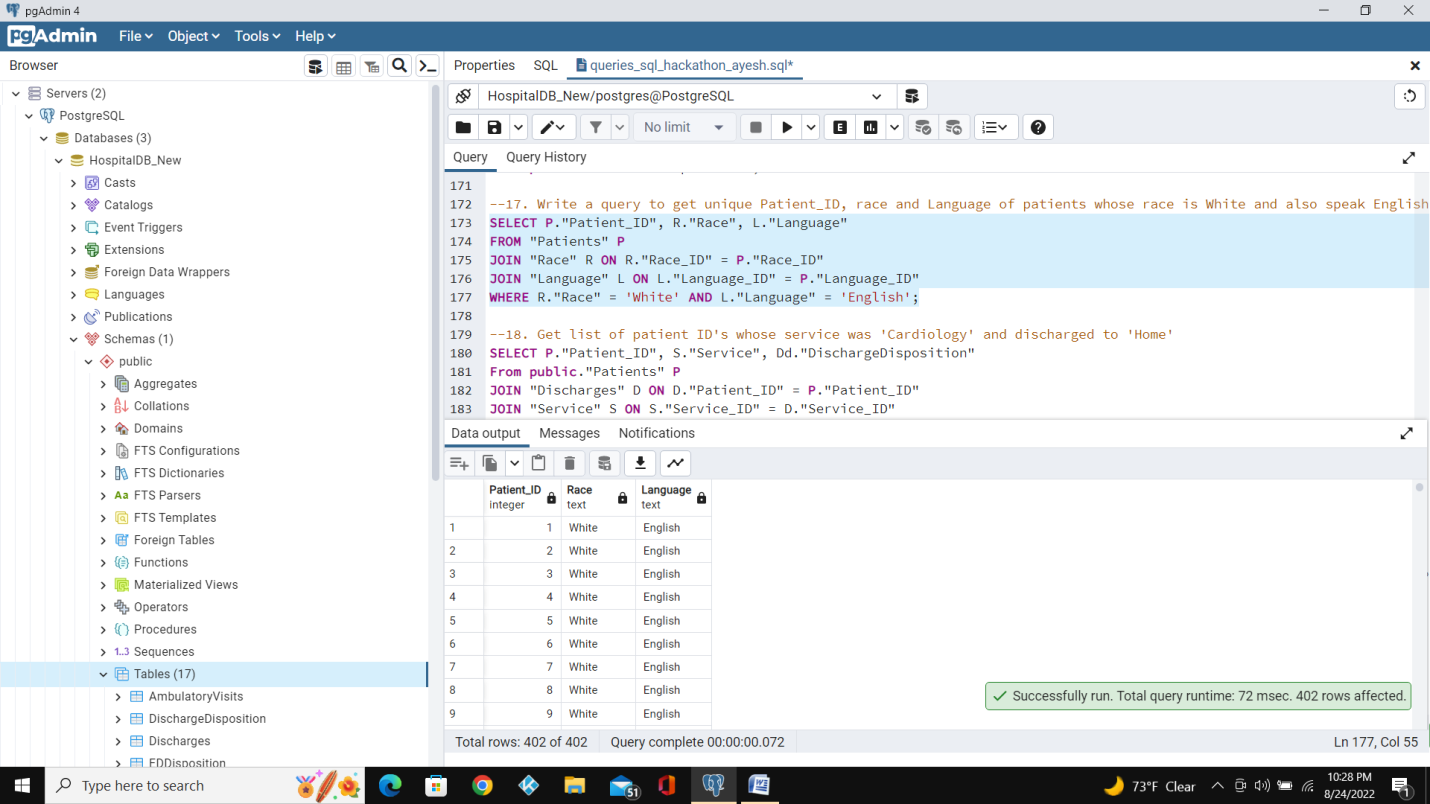
SELECT P."Patient\_ID", R."Race", L."Language"

FROM "Patients" P

Left JOIN "Race" R ON R."Race\_ID" = P."Race\_ID"

Left JOIN "Language" L ON L."Language\_ID" = P."Language\_ID"

WHERE R."Race" = 'White' AND L."Language" = 'English';



18. Get list of patient ID's whose service was 'Cardiology' and discharged to 'Home'.

Query:

SELECT P."Patient\_ID", S."Service", Dd."DischargeDisposition"

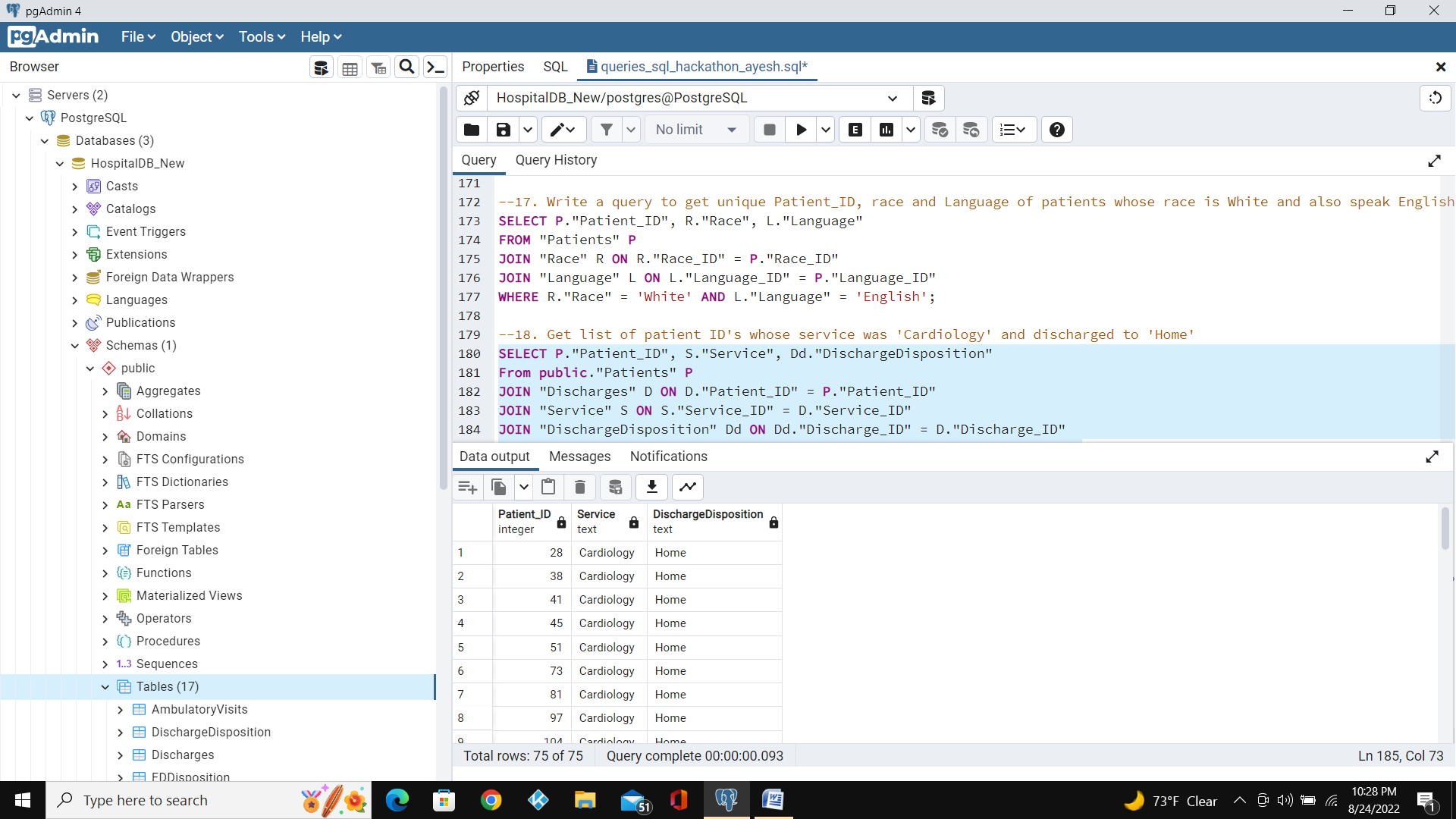
From public."Patients" P

LEFT JOIN "Discharges" D ON D."Patient\_ID" = P."Patient\_ID"

LEFT JOIN "Service" S ON S."Service\_ID" = D."Service\_ID"

LEFT JOIN "DischargeDisposition" Dd ON Dd."Discharge\_ID" = D."Discharge\_ID"

WHERE S."Service" = 'Cardiology' AND Dd."DischargeDisposition" = 'Home';

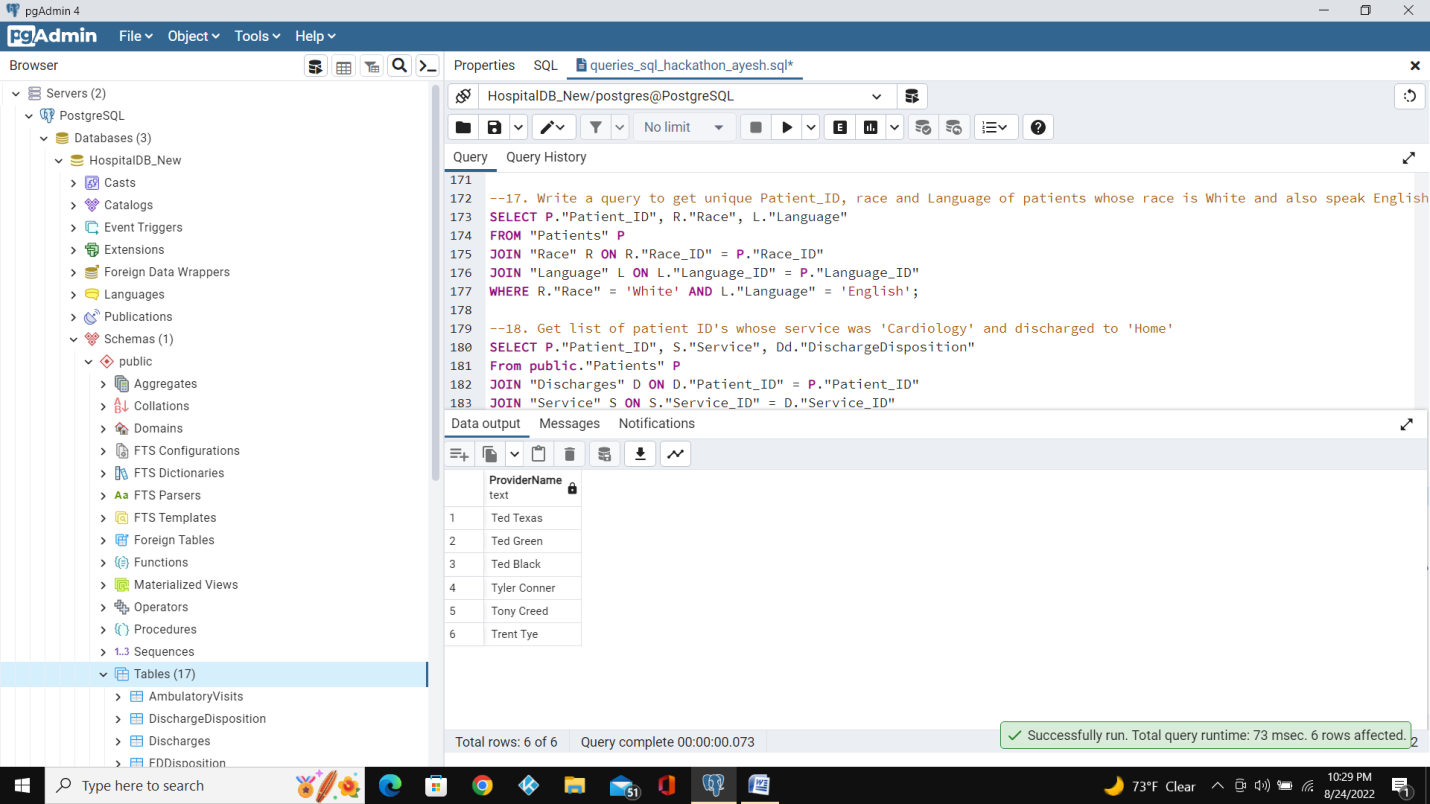


19. Write a query to get list of Provider names whose Providername is starting with letter T.

Query:

SELECT P."ProviderName" FROM public."Providers" P

WHERE "ProviderName" LIKE 'T%';



20. List female patients over the age of 40 who have undergone surgery from January-March 2019.

Query:

SELECT P."FirstName", P."LastName", age(P."DateOfBirth"), G."Gender", Ps."ProviderSpeciality", A."DateofVisit"

FROM "Patients" P

Left JOIN "Gender" G ON G."Gender\_ID" = P."Gender\_ID"

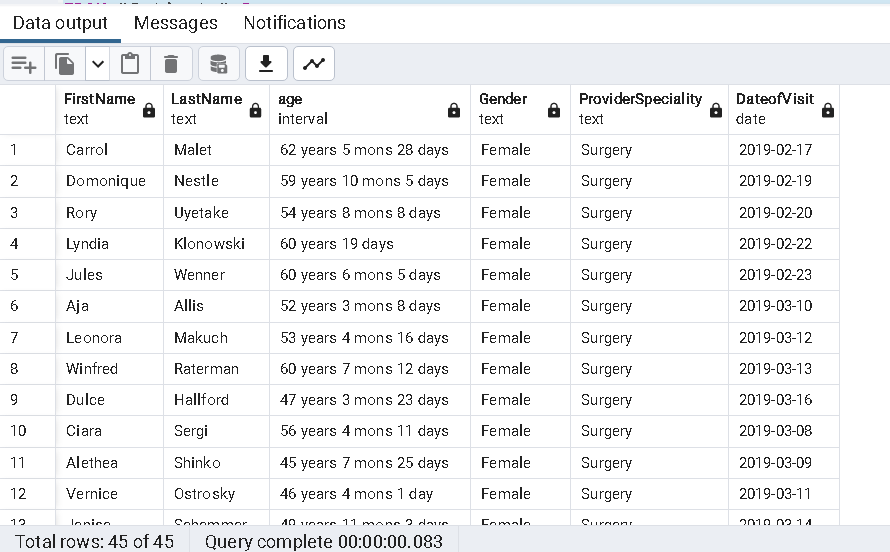
Left JOIN "AmbulatoryVisits" A ON A."Patient\_ID" = P."Patient\_ID"

Left JOIN "Providers" Pr ON Pr."Provider\_ID" = A."Provider\_ID"

Left JOIN "ProviderSpeciality" Ps ON Ps."PS\_ID" = Pr."PS\_ID"

WHERE (EXTRACT(YEAR FROM "DateofVisit") - EXTRACT(YEAR FROM "DateOfBirth")) > '40' AND G."Gender" = 'Female' AND Ps."PS\_ID" = 'PSID02'

AND A."DateofVisit" BETWEEN '2019-01-01' AND '2019-03-31';



21. Write a Query to get list of Male patients.

Query:

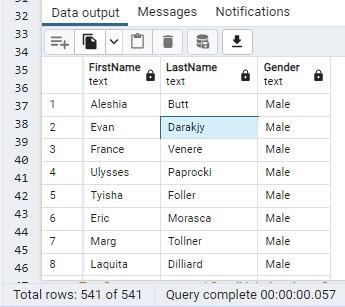
select "FirstName","LastName","Gender"

from public."Patients" pat

left outer join public."Gender" gen on gen."Gender\_ID" = pat."Gender\_ID"

where "Gender" = 'Male'

order by "Patient\_ID";



22. Write a query to get list of patient ID's who has discharged to home.

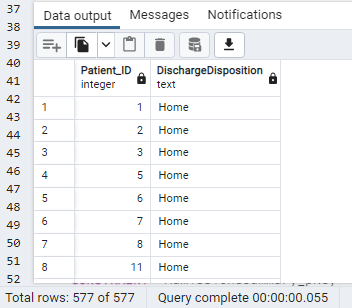
Query:

select DIS."Patient\_ID",disp."DischargeDisposition"

from public."Discharges" dis

left outer join public."DischargeDisposition" disp on dis."Discharge\_ID" = disp."Discharge\_ID"

where "DischargeDisposition" = 'Home';



23. Find the category of illness(Stomach Ache or Migrane) that has maximum number of patients.

Query:

select \* from (

select "ReasonForVisit",count(distinct "Patient\_ID") patient\_cnt,

rank() over(order by count(distinct "Patient\_ID") desc) patient\_cnt\_rnk

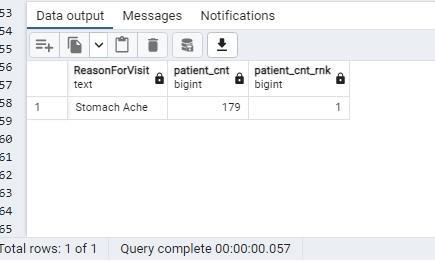
from public."EDVisits" vis

left outer join public."ReasonForVisit" res on res."Rsv\_ID" = vis."Rsv\_ID"

where ("ReasonForVisit" = 'Stomach Ache' or "ReasonForVisit" ='Migraine')

group by "ReasonForVisit" ) as pat\_vis

where patient\_cnt\_rnk = 1;



24. Write a query to get list of New Patient ID's.

Query:

select pat."Patient\_ID",vtyp."VisitType"

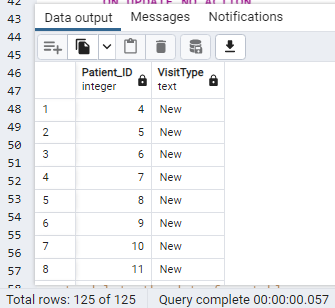
from public."Patients" pat

left outer join public."AmbulatoryVisits" vis on vis."Patient\_ID" = pat."Patient\_ID"

left outer join public."VisitTypes" vtyp on vtyp."AMVT\_ID" = vis."AMVT\_ID"

where "VisitType" = 'New'

order by pat."Patient\_ID";



25. Create trigger on table Readmission registry.

Explanation :To Cretae new table "AdmissionsSummary" in HospitalDB\_New

Query:

CREATE TABLE IF NOT EXISTS public."AdmissionsSummary"

("SummaryID" SERIAL,

"Patient\_ID" integer NOT NULL,

"AdmissionCount" integer,

CONSTRAINT "AdmissionsSummary\_pkey" PRIMARY KEY ("SummaryID", "Patient\_ID"),

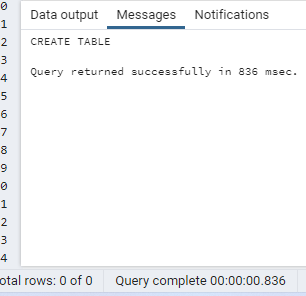
CONSTRAINT "Patient\_ID\_FK" FOREIGN KEY ("Patient\_ID")

REFERENCES public."Patients" ("Patient\_ID") MATCH SIMPLE

ON UPDATE NO ACTION

ON DELETE NO ACTION)

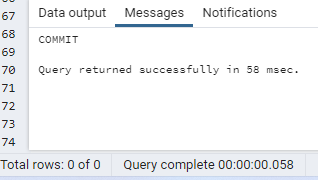
TABLESPACE pg\_default;



Explanation-- Commit

Query:

Commit;

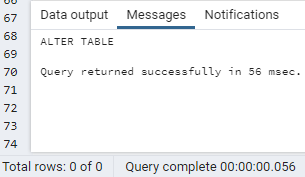


Explanation --Run ALTER

Query:

ALTER TABLE IF EXISTS public."AdmissionsSummary"

OWNER to postgres;



Explanation:Creating a Function and Setting Trigger On public."ReAdmissionRegistry".This trigger will Count total number of times the patient is admitted as soon as the new dataset is inserted in "ReAdmissionRegistry"

Query:

CREATE or replace FUNCTION add\_patient\_admission\_summary() RETURNS trigger AS $pat\_add$

DECLARE

v\_patient\_exists integer;

BEGIN

select count("Patient\_ID")

into v\_patient\_exists

from public."AdmissionsSummary"

where "Patient\_ID" = new."Patient\_ID";

if v\_patient\_exists = 1 then

update public."AdmissionsSummary"

set "AdmissionCount" = "AdmissionCount" + 1;

else

insert into public."AdmissionsSummary"

("Patient\_ID", "AdmissionCount")

values

(NEW."Patient\_ID",1) ;

end if;

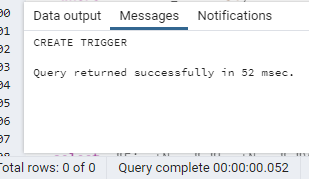
RETURN NULL; -- result is ignored since this is an AFTER trigger

END;

$pat\_add$ LANGUAGE plpgsql;

CREATE or replace TRIGGER pat\_add AFTER INSERT ON public."ReAdmissionRegistry"

FOR EACH ROW EXECUTE FUNCTION add\_patient\_admission\_summary();



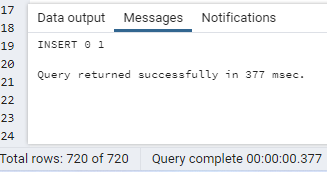
Explanation:Insert new data in "ReAdmissionRegistry" (Addeed same Patient Id twice with new Admission ID)

Query:

INSERT INTO public."ReAdmissionRegistry"(

"Admission\_ID", "Patient\_ID", "AdmissionDate", "DischargeDate", "Discharge\_ID", "Service\_ID", "Diagnosis\_ID", "ExpectedLOS", "ExpectedMortality", "ReadmissionFlag", "DaysToReadmission", "EDVisitAfterDischargeFlag")

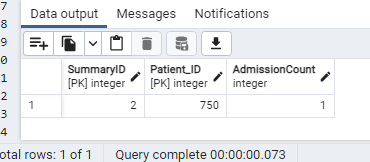
VALUES ('724', '750', '2018-03-02', '2018-01-13 15:59:00', 'DID02', 'SID06', 'PD015', '9.954144', '0.384839', '1', '2', '1');

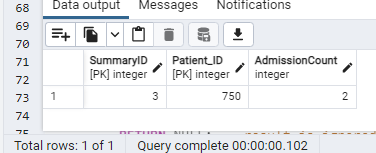


Explanation:To check if new record is added to "AdmissionsSummary";

Query:

select \* from public."AdmissionsSummary";





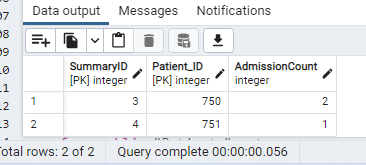
Explanation:Insert new data in "ReAdmissionRegistry" (Addeed New Patient Id with new Admission ID)

Query:

INSERT INTO public."ReAdmissionRegistry"(

"Admission\_ID", "Patient\_ID", "AdmissionDate", "DischargeDate", "Discharge\_ID", "Service\_ID", "Diagnosis\_ID", "ExpectedLOS", "ExpectedMortality", "ReadmissionFlag", "DaysToReadmission", "EDVisitAfterDischargeFlag")

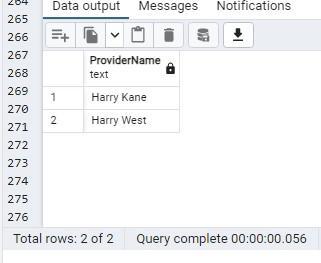
VALUES ('724', '751', '2018-03-02', '2018-01-13 15:59:00', 'DID02', 'SID06', 'PD015', '9.954144', '0.384839', '1', '2', '1');



26. Select all providers with a name starting 'h' followed by any character , followed by 'r', followed by any character,followed by 'y'

Query:

Select "ProviderName" from public."Providers" where "ProviderName" like 'H\_r\_y%';



27. Show the list of the patients who have cancelled their appointment.

Query:

select "FirstName","LastName","VisitStatus"

from public."Patients" pat

left outer join public."AmbulatoryVisits" avis on avis."Patient\_ID" = pat."Patient\_ID"

left join public."VisitStatus" stat on stat."VisitStatus\_ID" = avis."VisitStatus\_ID"

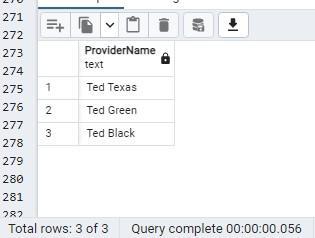
where stat."VisitStatus" = 'Canceled';



28. Write a query to get list of ProviderName's with a name starting 'ted'.

Query:

Select "ProviderName" from public."Providers" where "ProviderName" LIKE INITCAP('ted%')



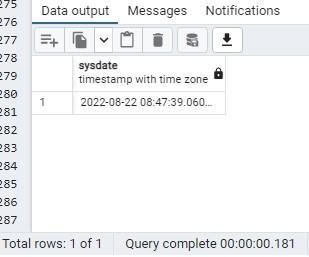
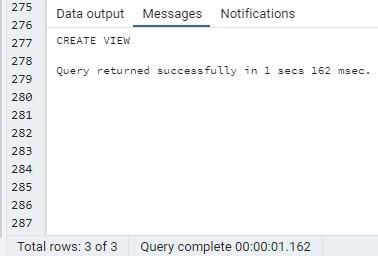
29. Create a view without using any schema or table and check the created view using select statement.

Query:

create or replace view getNow as

select now() as sysdate;

select \* from getNow;



30. Write a query to get unique list of Patient Id's whose reason for visit is car accident.

Query:

select distinct pat."Patient\_ID","ReasonForVisit"

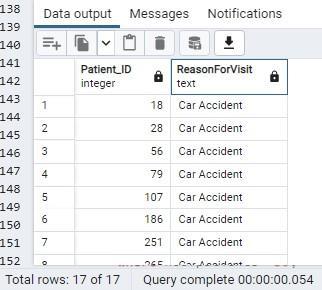
from public."Patients" pat

left join public."EDVisits" vis on vis."Patient\_ID" = pat."Patient\_ID"

left join public."ReasonForVisit" res on res."Rsv\_ID" = vis."Rsv\_ID"

where "ReasonForVisit" = 'Car Accident'

order by pat."Patient\_ID";



31. Find which Visit type of patients are maximum in cancelling their appointment

Query:

select \* from (

select "VisitType"

,"VisitStatus"

,Count("VisitStatus") cnt

,rank () over(order by Count("VisitStatus") desc ) cnt\_rnk

from public."AmbulatoryVisits" avis

left outer join public."VisitStatus" stat on stat."VisitStatus\_ID" = avis."VisitStatus\_ID"

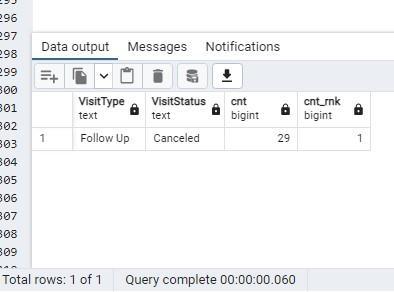
left outer join public."VisitTypes" typ on typ."AMVT\_ID" = avis."AMVT\_ID"

where "VisitStatus" = 'Canceled'

group by "VisitType" ,"VisitStatus"

order by cnt\_rnk ) viscnt

where cnt\_rnk = 1;



32. Write a query to Count number of patients by VisitdepartmentID where count greater than 50.

Query:

Select "VisitDepartmentID", noofpatients from (

select "VisitDepartmentID", COUNT(distinct pat."Patient\_ID") noofpatients

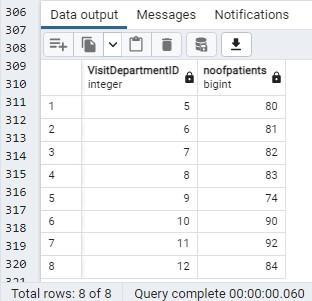
from public."Patients" pat

inner join public."AmbulatoryVisits" avis on avis."Patient\_ID" = pat."Patient\_ID"

Group by "VisitDepartmentID"

ORDER by "VisitDepartmentID") as patnum

where noofpatients >50;



33. Write a query to get list of patient names whose visit type is new and visitdepartmentId is 2.

Query:

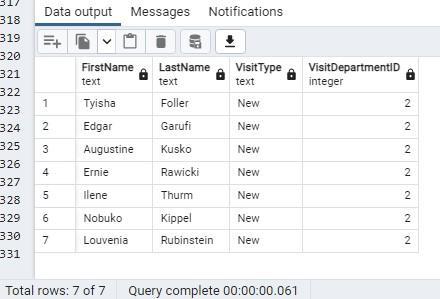
select "FirstName","LastName","VisitType","VisitDepartmentID"

from public."Patients" pat

left join public."AmbulatoryVisits" avis on avis."Patient\_ID" = pat."Patient\_ID"

left join public."VisitTypes" typ on typ."AMVT\_ID" = avis."AMVT\_ID"

where typ."VisitType" = 'New' and avis."VisitDepartmentID"=2;



34. Write a query to find the most common reasons for hospital visit for patients between 50 and 60 years.

Query:

select \* from (

select "ReasonForVisit"

,count("ReasonForVisit") visit\_cnt\_for\_50and60\_years

,rank() over(order by count(pat."Patient\_ID") desc) visit\_rnk

from public."Patients" pat

left join public."EDVisits" vis on vis."Patient\_ID" = pat."Patient\_ID"

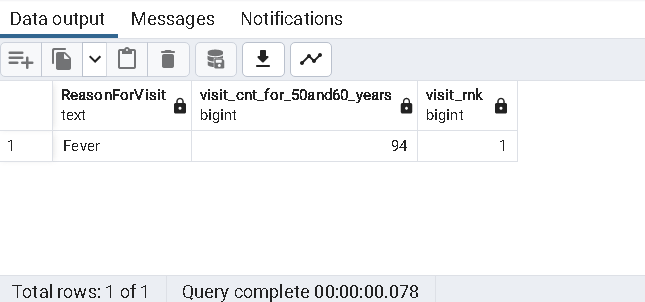
left join public."ReasonForVisit" rv on vis."Rsv\_ID"= rv."Rsv\_ID"

where (EXTRACT(YEAR FROM "VisitTimestamp") - EXTRACT(YEAR FROM "DateOfBirth")) between 50 and 60

group by "ReasonForVisit"

order by visit\_cnt\_for\_50and60\_years desc) res\_for\_vis

where visit\_rnk = 1;



35. Get list of Patients whose gender is Male and who speak English and whose race is White.

Query:

select "FirstName","LastName","Gender","Language","Race"

from public."Patients" pat

left join public."Language" lan on lan."Language\_ID" = pat."Language\_ID"

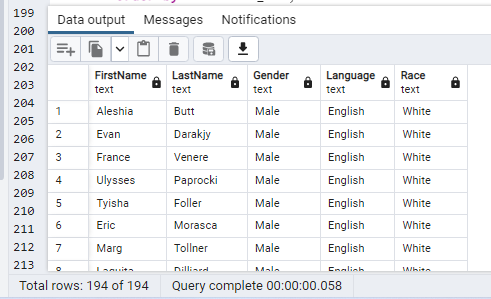
left join public."Race" rac on rac."Race\_ID" = pat."Race\_ID"

left join public."Gender" gen on gen."Gender\_ID" = pat."Gender\_ID"

where lan."Language" = 'English'

and "Race" = 'White'

and "Gender" = 'Male';



36. Create index on Patient table.

Query:

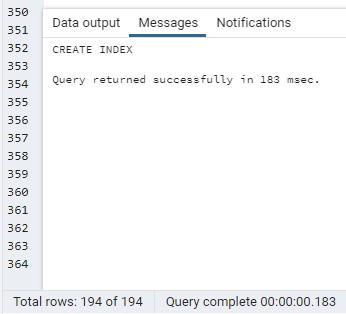
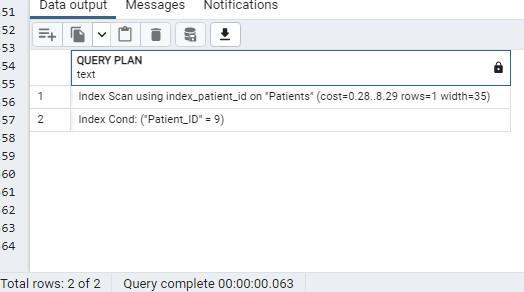
Create index index\_patient\_id

ON public."Patients"("Patient\_ID");

EXPLAIN SELECT \*

FROM public."Patients"

WHERE "Patient\_ID" = '9';

37. Write a query to get list of Provider ID's where ProviderDateOnStaff year is 2013 and 2010.

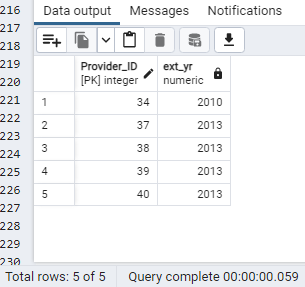
Query:

Select "Provider\_ID", EXTRACT(YEAR FROM "ProviderDateOnStaff") ext\_yr

from public."Providers"

where EXTRACT(YEAR FROM "ProviderDateOnStaff") in (2013,2010)

order by "Provider\_ID";



38. Write a query to find out percentage of Ambulatory visits by visit type.

Query:

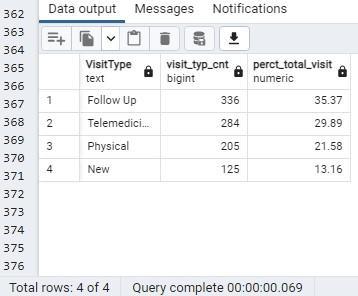
select typ."VisitType",count(\*) visit\_typ\_cnt, round(count(\*)/sum(count(\*)) over() \* 100,2) Perct\_total\_visit

from public."AmbulatoryVisits" avis

left join public."VisitTypes" typ on typ."AMVT\_ID" = avis."AMVT\_ID"

group by typ."VisitType"

order by visit\_typ\_cnt desc;



39. Write a query to get list of patient names who has discharged.

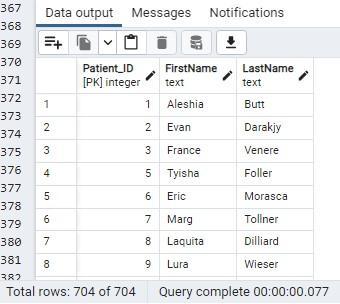
Query:

Select distinct pat."Patient\_ID", "FirstName","LastName"

from public."Patients" as pat

join public."Discharges" as dis On pat."Patient\_ID" = dis."Patient\_ID"

order by pat."Patient\_ID";

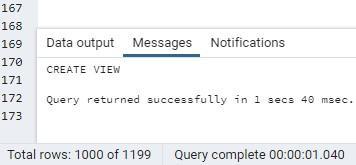


40. Create view on table EdVisit by selecting some columns and filter data using Where condition

Explanation:Part 1:

Query:

Create View EDV AS Select "Patient\_ID", "Acuity","EDD\_ID" from public."EDVisits";



Explanation:Part2:

Query:

Select \* from public.edv where "Acuity" = '2';



41. Get list of patient names whose primary diagnosis as 'Spinal Cord injury' having Expected LOS is greater than 15

Query:

Select PT."FirstName",PT."LastName",N."ExpectedLOS",N."PrimaryDiagnosis" from

(Select R."Patient\_ID",R."Diagnosis\_ID",R."ExpectedLOS",P."PrimaryDiagnosis" from

(Select r."Patient\_ID",r."Diagnosis\_ID",r."ExpectedLOS" from "ReAdmissionRegistry" r

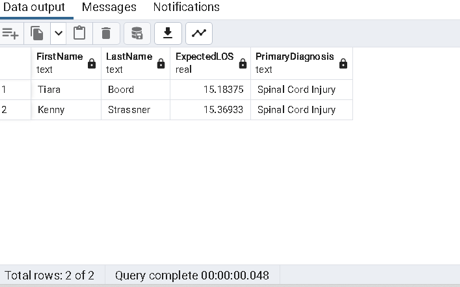
Where r."ExpectedLOS" > '15') R

join (Select "Diagnosis\_ID","PrimaryDiagnosis" from "PrimaryDiagnosis" Where "PrimaryDiagnosis" = 'Spinal Cord Injury') P

ON R."Diagnosis\_ID" = P."Diagnosis\_ID") N

Join "Patients" PT

ON N."Patient\_ID" = PT."Patient\_ID"

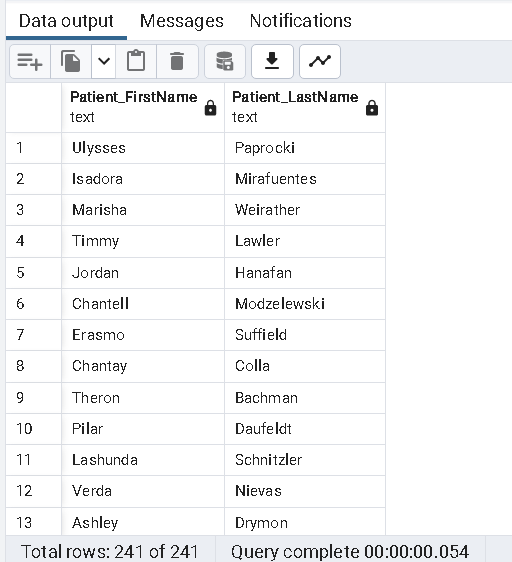


42. Write a query to get list of Patient names who haven't discharged.

Query:

Select P."FirstName" as "Patient\_FirstName",P."LastName" as "Patient\_LastName" from "Patients" P

Where P."Patient\_ID" NOT IN (Select D."Patient\_ID" from "Discharges" D);



43. Write a query to get list of Provider names whose ProviderSpecialty is Pediatrics.

Query:

Select Pr."ProviderName", PS."ProviderSpeciality" from "Providers" Pr

left join public."ProviderSpeciality" PS ON Pr."PS\_ID" = PS."PS\_ID"

Where "ProviderSpeciality" = 'Pediatrics';

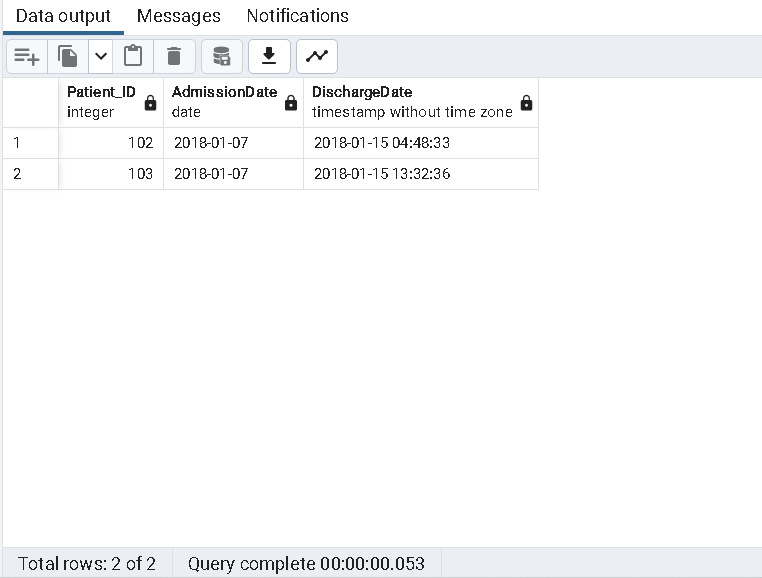


44. Write a query to get list of patient ID's who has admitted on 1/7/2018 and discharged on 1/15/2018

Query:

Select D."Patient\_ID",D."AdmissionDate",D."DischargeDate" from "Discharges" D

Where D."AdmissionDate" ='2018-01-07' AND D."DischargeDate"::timestamp::date ='2018-01-15';



45. Write a query to find outpatients vs inpatients by monthwise (hint: consider readmission/discharges and ambulatory visits table for inpatients and outpatients)

Query:

Select T1."OutPatients",T2."InPatients",T1."InMonthOf"

from (Select count(public."AmbulatoryVisits"."Patient\_ID") as "OutPatients",TO\_CHAR("AmbulatoryVisits"."DateofVisit",'Month') as "InMonthOf" from public."AmbulatoryVisits"

Where "VisitStatus\_ID" = 'VS002' and public."AmbulatoryVisits"."Patient\_ID"

not in (select public."Discharges"."Patient\_ID" from public."Discharges")

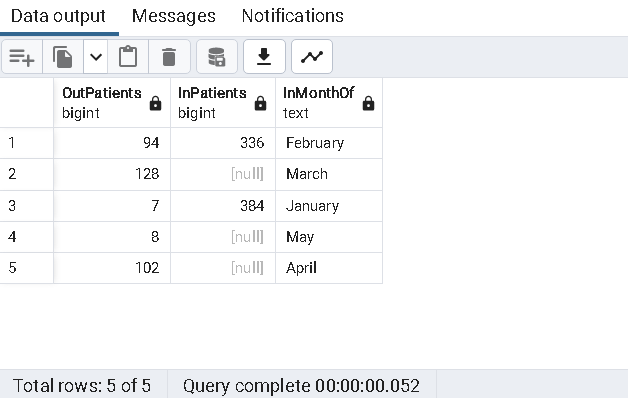
Group by "InMonthOf") T1

Left Join (SELECT Count("Patient\_ID") as "InPatients",TO\_CHAR("AdmissionDate",'Month') AS "InMonthOf"

from "Discharges"

Group by "InMonthOf") as T2

on T1."InMonthOf" = T2."InMonthOf";



46. Write a query to get list of Number of Ambulatory Visits by Provider Speciality per month

Query:

Select TO\_CHAR(avis."DateofVisit",'Month') as Mnth , PS."ProviderSpeciality", Count(avis."AMVT\_ID") as "NoOfAmbulatoryVisit" from public."AmbulatoryVisits" avis

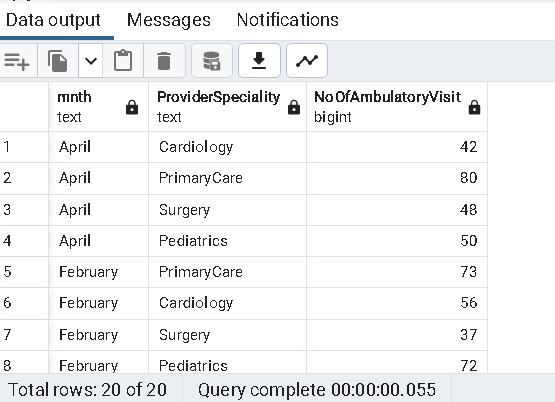
left join public."VisitTypes" vis on avis."AMVT\_ID" = vis."AMVT\_ID"

left join public."Providers" P on avis."Provider\_ID" = P."Provider\_ID"

left join public."ProviderSpeciality" PS on PS."PS\_ID"=P."PS\_ID"

Group by Mnth,PS."ProviderSpeciality"

Order by Mnth;



47. Write a query to find Average age for admission by service

Query:

Select Ser."Service",AVG(AGE(Ser."AdmissionDate",Ser."DateOfBirth")) as "AvrgAgeOfPatientsAdmitted"

from (Select S."Patient\_ID",S."Service",S."AdmissionDate", PT."DateOfBirth" from

(Select R."Patient\_ID", R."Service\_ID",S."Service",R."AdmissionDate" from "ReAdmissionRegistry" R

Join "Service" S

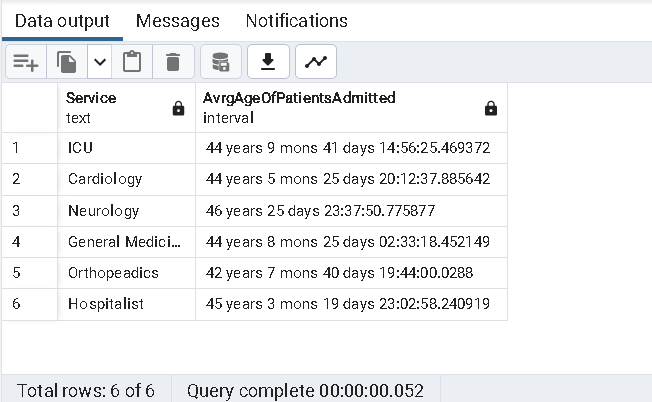
ON R."Service\_ID" = S."Service\_ID") S

Join(Select P."Patient\_ID",P."DateOfBirth"

from "Patients" P) PT

on S."Patient\_ID" = PT."Patient\_ID") Ser

Group by Ser."Service"



48. Write a query to get list of patient with their full names whose names contains "Ma"

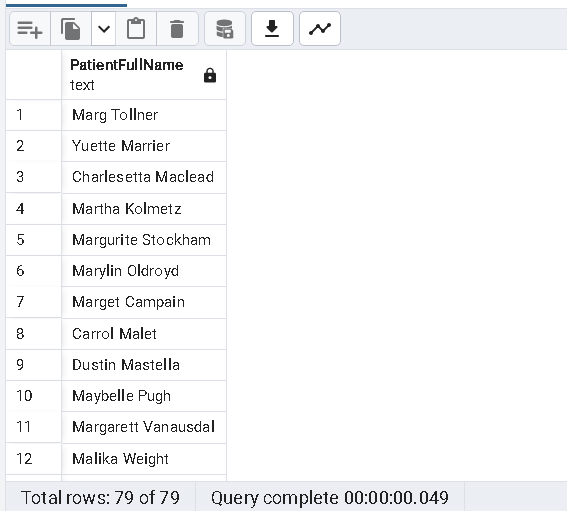
Query:

Select P."PatientFullName"

from (Select concat("FirstName",' ',"LastName") AS "PatientFullName"

From "Patients") P

where P."PatientFullName" Like '%Ma%'



49. Update Visit Timestamp column in EDVisits table by selecting data type as timestamp with timezone

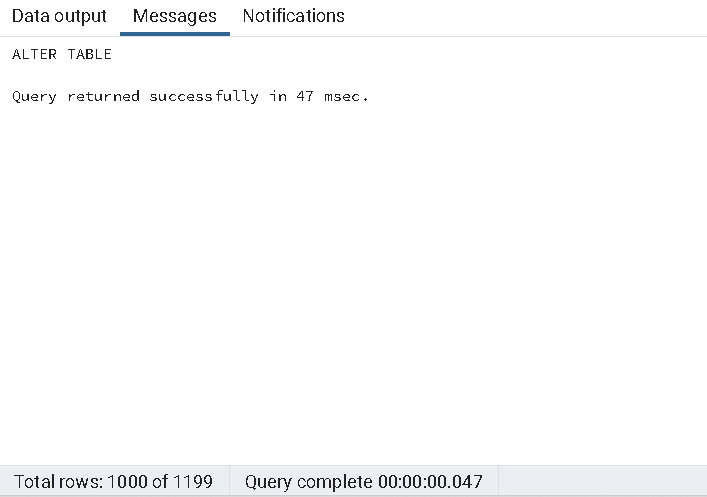
Query:

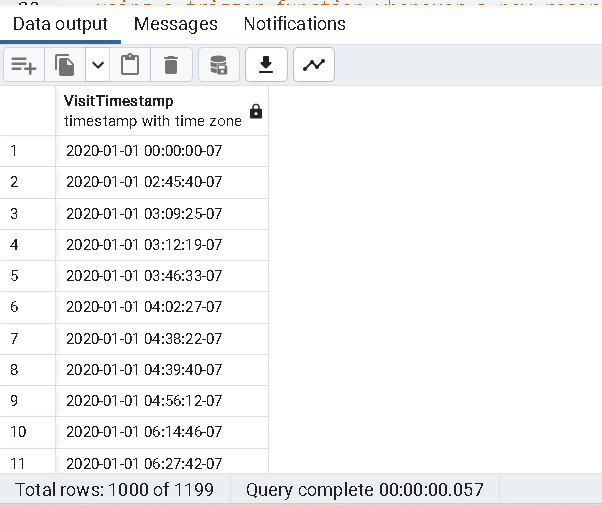
alter table "EDVisits" alter column "VisitTimestamp" type timestamp with time zone;

Explanation:Query to view the altered column.

Query:

select "VisitTimestamp" from "EDVisits"





50. Write a create a trigger function on AmbulatoryVisits by selecting any two columns.

Explanation: Query to Creating a Table "AmbulatoryVisitsPatientBPLog" to log the patients Blood pressure details

Query:

CREATE TABLE IF NOT EXISTS public."AmbulatoryVisitsPatientBPLog"

(

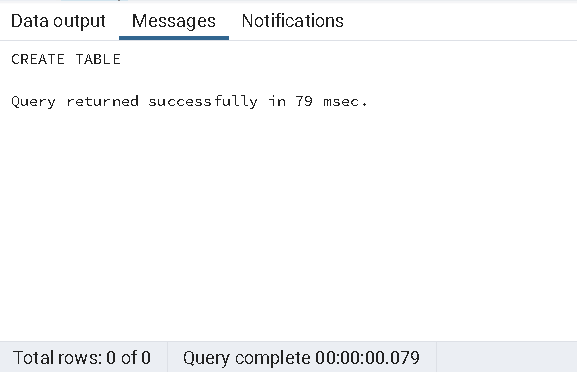
"Patient\_ID" integer,

"BloodPressureSystolic" real,

"BloodPressureDiastolic" real

)

TABLESPACE pg\_default;



Explanation: Query to alter created table:

Query:

ALTER TABLE IF EXISTS public."AmbulatoryVisitsPatientBPLog"

OWNER to postgres;



Explanation: Query to display the newly created table "AmbulatoryVisitsPatientBPLog":

Query:

Select \* from public."AmbulatoryVisitsPatientBPLog"

Explanation: Creating a Trigger Function to be called when trigger is invoked to insert the records in the newly created "AmbulatoryVisitsPatientBPLog" table

Query:

CREATE OR REPLACE FUNCTION PatientBP\_insert()

RETURNS trigger AS $Insert\_BP$

BEGIN

INSERT INTO public."AmbulatoryVisitsPatientBPLog"("Patient\_ID","BloodPressureSystolic","BloodPressureDiastolic")

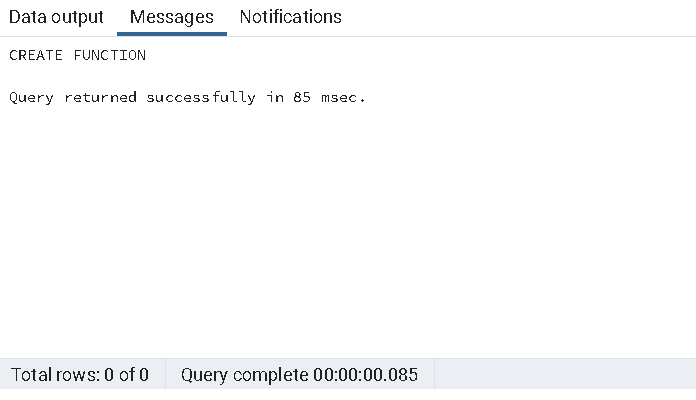
VALUES(NEW."Patient\_ID",NEW."BloodPressureSystolic",New."BloodPressureDiastolic");

RETURN Null;

END;

$Insert\_BP$

LANGUAGE 'plpgsql';



Explanation: Creating the Trigger Insert\_BP to insert the bp records to the "AmbulatoryVisits" Table

Query:

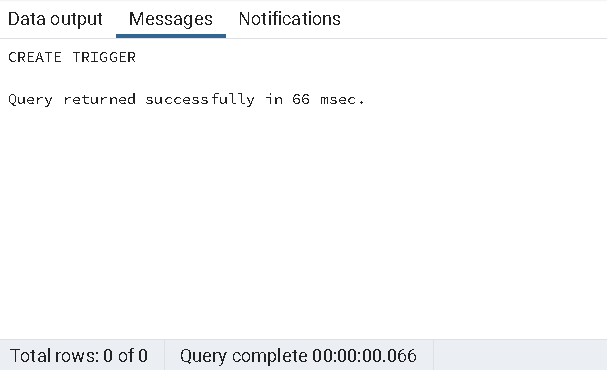
CREATE or replace TRIGGER Insert\_BP

AFTER INSERT

ON public."AmbulatoryVisits"

FOR EACH ROW

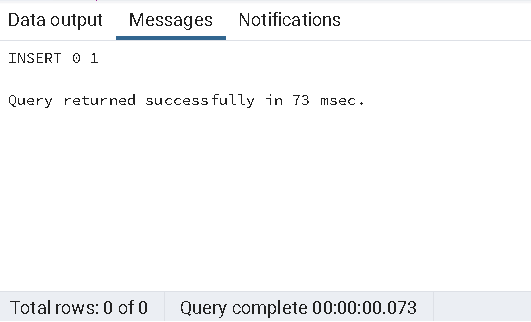
EXECUTE FUNCTION PatientBP\_insert();



Explanation: Inserting new record into table public."AmbulatoryVisits"

Query:

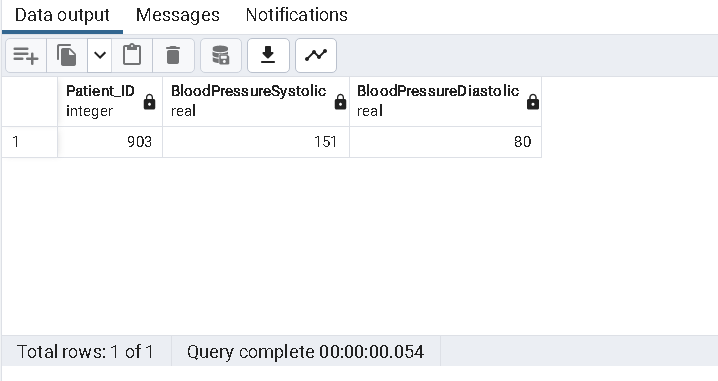
INSERT INTO public."AmbulatoryVisits" VALUES (952,102,1,'2019-02-18','2019-01-24 11:50:26',12,'AMVT002',157,95,90,'VS003')



Explanation: Displaying the new table with records inserted successfully with the created trigger

Query:

Select \* from public."AmbulatoryVisitsPatientBPLog"



51. Insert number of days for Readmission in DaysToReadmission Column for patient ID's from 737 to 742 .( Use any random value)

Query:

Update "ReAdmissionRegistry" set "DaysToReadmission" =

CASE

WHEN "Patient\_ID" = '737' THEN 7

WHEN "Patient\_ID" = '738' THEN 8

WHEN "Patient\_ID" = '739' THEN 9

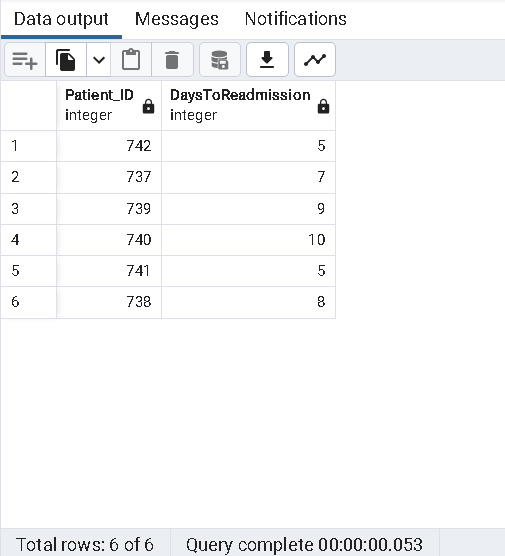
WHEN "Patient\_ID" = '740' THEN 10

ELSE 5

END

where "Patient\_ID" BETWEEN 737 and 742

Returning "Patient\_ID","DaysToReadmission"

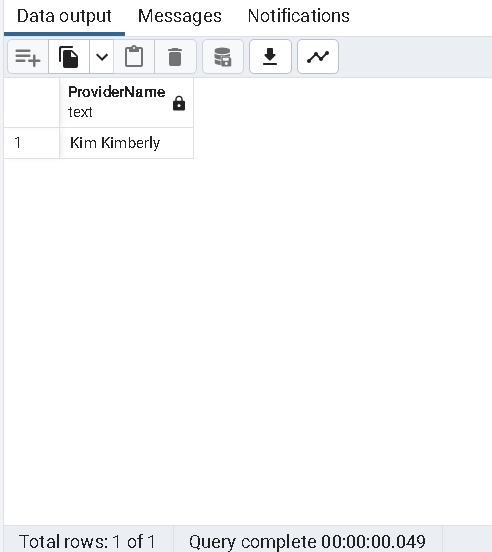


52. Get list of Provider names whose name is starting with K and ending with y (Hint:K-Upper, Y-Lower)

Query:

Select "ProviderName" from "Providers" P

Where P."ProviderName" LIKE 'K%y'



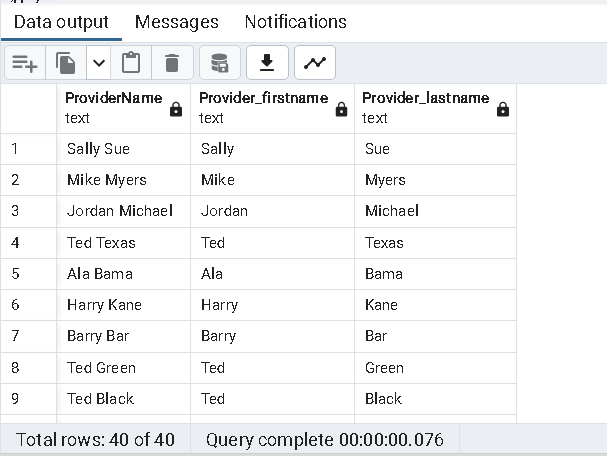
53. Write a query to Split provider First name and Last name into different column.

Query:

Select "ProviderName", split\_part("ProviderName",' ',1) as "Provider\_firstname",

split\_part("ProviderName",' ',2) as "Provider\_lastname"

from "Providers"

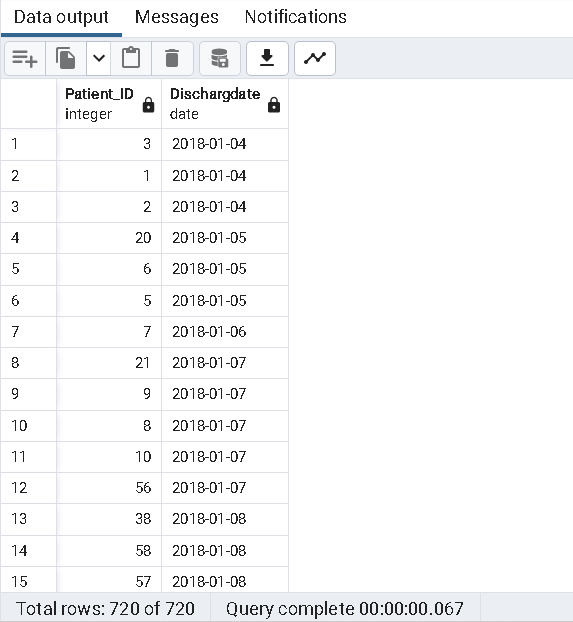


54. Get list of Patient ID's order by Discharge date

Query:

Select "Patient\_ID",(D."DischargeDate"::timestamp::date) AS "Dischargdate" from "Discharges" D

order by (D."DischargeDate"::timestamp::date)



55. Write a query to drop View by creating view on table Discharge by selecting columns

Explanation:Query to create a View from Discharge Table

Query:

Create View DischargeTableView

as

Select

"Admission\_ID",

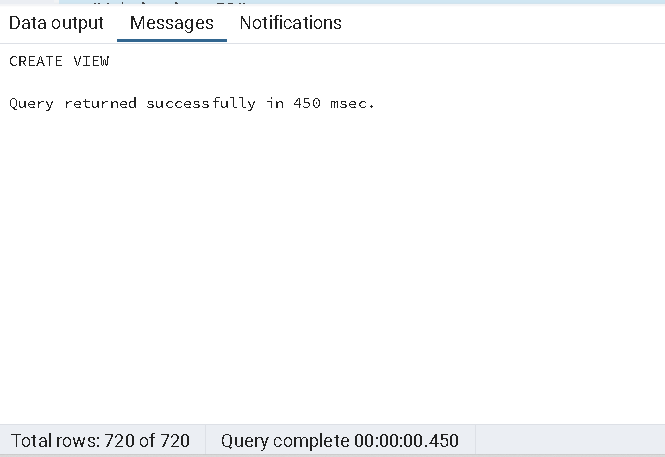
"Patient\_ID",

"DischargeDate"

from

"Discharges"

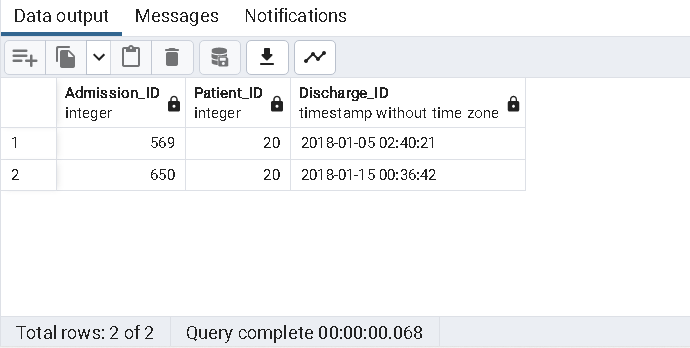
Where "Patient\_ID" ='20'



**Explanation:Query to view the created View**

**Query:**

Select \* from DischargeTableView

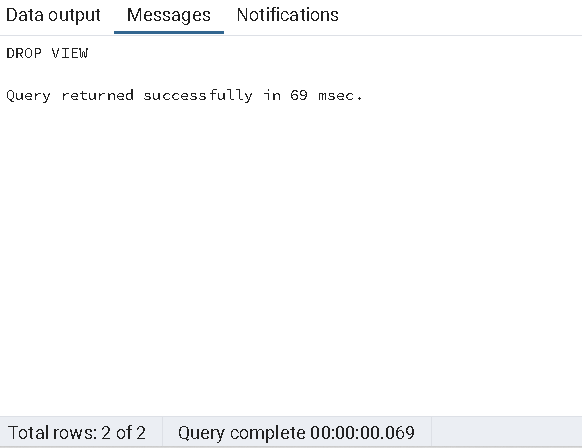


Explanation: Query to Drop the Created View

Query:

Drop View DischargeTableView

.



56. Write a query to get list of Patient ID's where Visitdepartment ID is 1 and-- BloodPressureSystolic is between 123 to 133

Query:

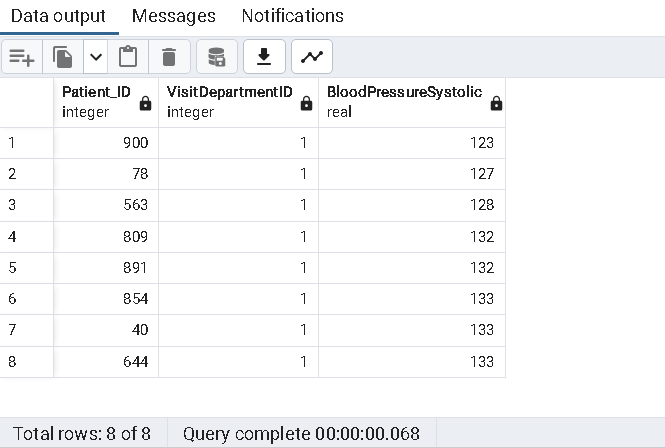
Select "Patient\_ID","VisitDepartmentID","BloodPressureSystolic"

from "AmbulatoryVisits" A

Where A."VisitDepartmentID"= '1'and

A."BloodPressureSystolic" Between 123 and 133

order by A."BloodPressureSystolic" ASC



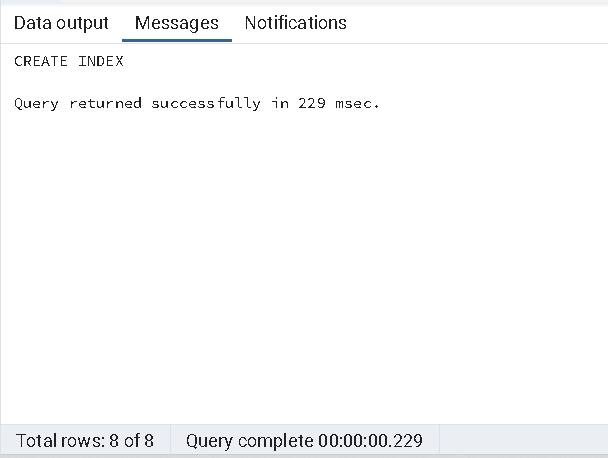
57. Write the query to create Index on table ReasonForVisit by selecting a column and also write the query drop same index

Explanation:Query To create an index for the values in the "ReasonForVisit" column from "ReasonforVisit" table

Query:

CREATE INDEX Reasonforvisitindex

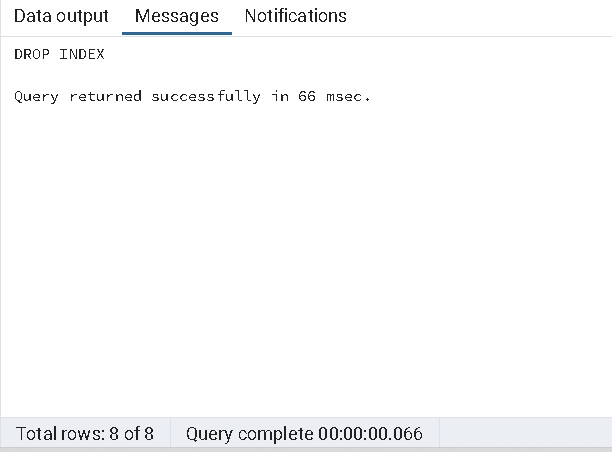
ON "ReasonForVisit"("ReasonForVisit")



Explanation: Query to drop an created index

Query:

DROP INDEX Reasonforvisitindex



58. Write a query to Count number ofunique patients EDDisposition wise.

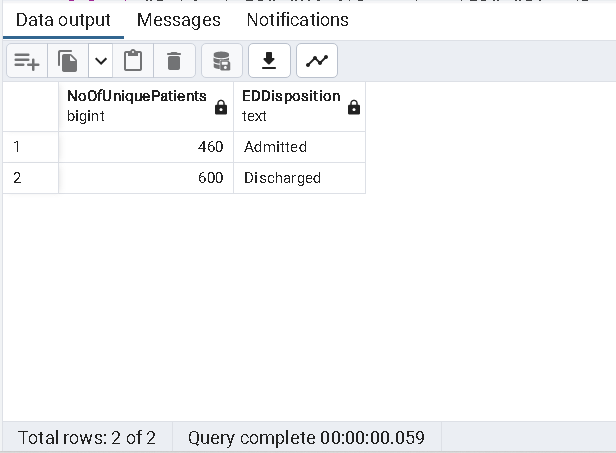
Query:

select count (Distinct evis."Patient\_ID") as "NoOfUniquePatients",edis."EDDisposition"

from public."EDVisits" evis

left join public."EDDisposition" edis on edis."EDD\_ID"=evis."EDD\_ID"

Group by edis."EDDisposition";



59. Write a query to get list of Patient ID's where Visitdepartment ID is 5 or-- BloodPressureSystolic is NOT NULL

Query:

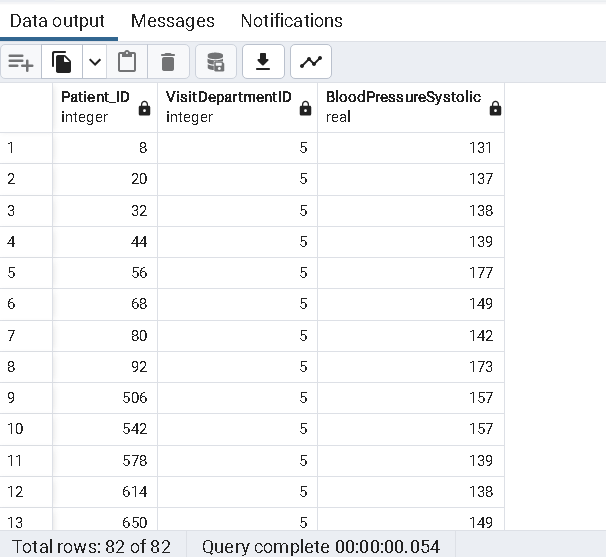
Select "Patient\_ID","VisitDepartmentID","BloodPressureSystolic"

from "AmbulatoryVisits" A

Where A."VisitDepartmentID"= '5'

or

A."BloodPressureSystolic" IS NOT NULL;



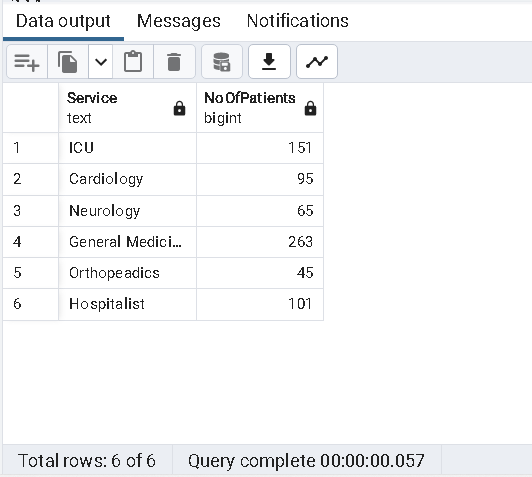
60. Query to find the number of patients readmitted by Service

Query:

Select "Service",Count("Patient\_ID") as "NoOfPatients" from public."ReAdmissionRegistry" rad

left join public."Service" S on S."Service\_ID" = rad."Service\_ID"

Group By "Service";



61. Write a query to list male patient ids and their names who are above 40

-- years of age and less than 60 years and have BloodPressureSystolic above

-- 120 and BloodPressureDiastolic above 80.

Query:

select p."Patient\_ID",p."FirstName",p."LastName",g."Gender",age(NOW(),"DateOfBirth"),

A."BloodPressureSystolic",A."BloodPressureDiastolic"

from "Patients" as p Left Join "Gender" as g

on p."Gender\_ID" = g."Gender\_ID"

Left Join "AmbulatoryVisits" as A

on p."Patient\_ID" = A."Patient\_ID"

where "Gender" IN ('Male') AND

"BloodPressureSystolic" > 120 AND

"BloodPressureDiastolic" > 80 AND

Extract(Years from age(NOW(),"DateOfBirth")) Between 40 AND 59;

Graphical user interface, table

Description automatically generated

62. Query to find the number of patients who have visited month wise.

Query:

select NoOfPatientsVisited, InMonthOf

from (select count("Patient\_ID")as NoOfPatientsVisited,TO\_CHAR("DateofVisit",'Month') as InMonthOf,

Extract(Month from "DateofVisit") as MonthNum

from "AmbulatoryVisits" Group by InMonthOf, MonthNum )as subtable

order by MonthNum

Graphical user interface, table

Description automatically generated

63. Write a query to get list of patient ID's whose BloodPressureSystolic is 131,137,138

Query:

select p."Patient\_ID",A."BloodPressureSystolic"

from "Patients" as p

Left Join "AmbulatoryVisits" as A

on p."Patient\_ID" = A."Patient\_ID"

where "BloodPressureSystolic" IN (131,137,138);

Table

Description automatically generated

1. Query to classify expected LOS into 3 categories as per the duration. (Hint: Use of CASE statement)

*Explanation:On Discharges table.*

Query:

select "ExpectedLOS",

CASE

WHEN "ExpectedLOS" BETWEEN 2 and 6 THEN '1\_Minor'

WHEN "ExpectedLOS" BETWEEN 6 and 11 THEN '2\_Major'

WHEN "ExpectedLOS" BETWEEN 11 and 17 THEN '3\_Critical'

END "Severity"

from "Discharges"

order by "ExpectedLOS"

Graphical user interface, application

Description automatically generated

Explanation: *On ReAdmissionRegistry Table*

Query:

select "ExpectedLOS",

CASE

WHEN "ExpectedLOS" BETWEEN 2 and 6 THEN '1\_Minor'

WHEN "ExpectedLOS" BETWEEN 6 and 11 THEN '2\_Major'

WHEN "ExpectedLOS" BETWEEN 11 and 17 THEN '3\_Critical'

END "Severity"

from "ReAdmissionRegistry"

order by "ExpectedLOS"

Table

Description automatically generated

65. Write a query to create a table to list the names of patients whose date of

        birth is later than 1st jan 1960.Name the table as “Persons”.

Query:

CREATE TABLE “Persons” as

select concat("FirstName",' ',"LastName") as PatientName,"DateOfBirth" from "Patients"

where "DateOfBirth" > '1960-01-01'

select \* from “Persons”

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

66.Write a query to Count number of patients who has discharged after march3rd 2018.

Query:

select count("Patient\_ID") as "NoofPatientsDischargedAfter3rdMar2018" from "Discharges"

Where DATE("DischargeDate") > '2018-03-03'

Graphical user interface, application, Word

Description automatically generated

67. Replace ICU with emergency (Hint: Do not update or alter the table)

Query:

Select REPLACE("Service",'ICU','emergency') as "ServiceUpdated","Service\_ID" from "Service"

Table

Description automatically generated

68. Write a query to get Sum of ExpectedLOS for Service\_ID 'SID01'.

*Explanation:On Discharges table*

Query:

select sum("ExpectedLOS") from "Discharges" where "Service\_ID"='SID01'

**Graphical user interface, text, application, chat or text message, email

Description automatically generated**

*Explanation:On ReAdmissionRegistry Table.*

Query:

select sum("ExpectedLOS") from "ReAdmissionRegistry" where "Service\_ID"='SID01'

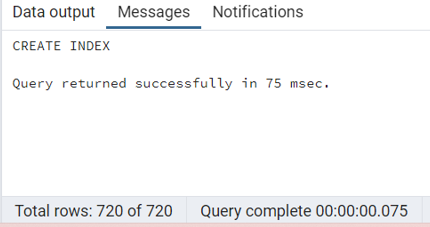
Graphical user interface, text, application, email

Description automatically generated

69. Create index on table Provider by selecting a column and filter by using WHERE condition.

Query:

Create INDEX "Index\_name\_Provider" On "Providers" ("ProviderName")



Query:

select \* from "Providers" where "ProviderName" IN ('Ted Black')

Graphical user interface, application, Word

Description automatically generated

70. List down all triggers in our HealthDB database.

Query:

SELECT  event\_object\_table AS table\_name ,trigger\_name

FROM information\_schema.triggers

GROUP BY table\_name , trigger\_name

ORDER BY table\_name ,trigger\_name

Graphical user interface, text, application, email

Description automatically generated

71. Partition the table according to Service\_ID and use windows function to calculate percent rank.   Order by ExpectedLOS.

Query:

select "Patient\_ID","Service\_ID","ExpectedLOS",

PERCENT\_RANK() OVER(partition by "Service\_ID" order by "ExpectedLOS") as "Percent\_Rank"

from "Discharges"

Table

Description automatically generated

Query:

select "Patient\_ID","Service\_ID","ExpectedLOS",

PERCENT\_RANK() OVER(partition by "Service\_ID" order by "ExpectedLOS") as "Percent\_Rank"

from "ReAdmissionRegistry"

Table

Description automatically generated

72. Write a query by using common table expressions and case statements to

-- display birthyear ranges.

Query:

WITH "cte\_PatientsAgeGroup" As (

select "Patient\_ID","FirstName","LastName","DateOfBirth",

CASE

WHEN Extract(Year from "DateOfBirth") <= 1966 THEN 'OlderAdulthood'

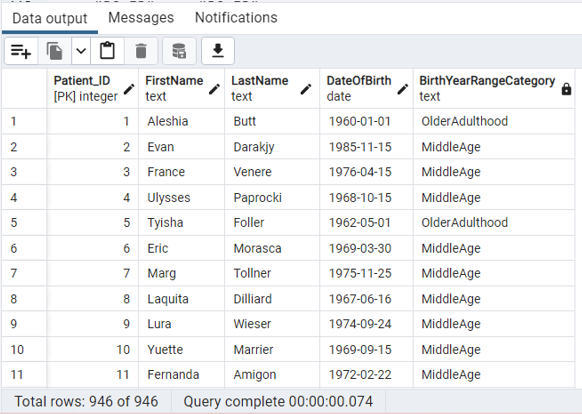
WHEN Extract(Year from "DateOfBirth") BETWEEN 1965 and 1986 THEN 'MiddleAge'

WHEN Extract(Year from "DateOfBirth") BETWEEN 1987 and 2004 THEN 'YoungAdulthood'

END "BirthYearRangeCategory"

from "Patients")

select \* from "cte\_PatientsAgeGroup";



73. Get list of Provider names whose ProviderSpeciality is Surgery.

Query:

select p."ProviderName",ps."ProviderSpeciality"

from "Providers" as p Left Join "ProviderSpeciality" as ps

on p."PS\_ID" =ps."PS\_ID"

where ps."ProviderSpeciality" = 'Surgery';

Graphical user interface

Description automatically generated

74. List of patient from rows 11-20 without using where condition.

Query:

select "Patient\_ID","FirstName","LastName" from "Patients" offset 10 rows Fetch next 10 rows only

Table

Description automatically generated

75. Give a query how to find triggers from table AmbulatoryVisits.

Query:

SELECT  event\_object\_table AS table\_name ,trigger\_name

FROM information\_schema.triggers

WHERE event\_object\_table = 'AmbulatoryVisits'

GROUP BY table\_name , trigger\_name

ORDER BY table\_name ,trigger\_name

Graphical user interface, text, application

Description automatically generated

76. Recreate the below expected output using Substring.

Query:

select "Gender",SUBSTRING("Gender",1,1) as "gender" from "Gender"

Graphical user interface, text, application

Description automatically generated

77. Obtain the below output by grouping the patients.

Query:

select "Patient\_ID","FirstName",('L') as "patient\_group" from "Patients"

Where "FirstName" LIKE 'L%'

Table

Description automatically generated

78. Please go through the below screenshot and create the exact output.

Query:

select "FirstName",char\_length("FirstName") as "LengthOfFirstName" from "Patients"

Table

Description automatically generated

79. Please go through the below screenshot and create the exact output BloodPressureDiastolic,pulse,bpd,heartrate.

Query:

select "BloodPressureDiastolic","Pulse",

       trunc("BloodPressureDiastolic"+1) as "bpd",trunc("Pulse") as "HeartRate"

from "AmbulatoryVisits" offset 1 row Fetch next 21 rows only

Table

Description automatically generated

80. Please go through the below screenshot and create the exact output

    string and numeric.

Query:

select "BloodPressureSystolic",

'The Systolic Blood pressure is '|| to\_char("BloodPressureSystolic",'999.99') as "Message"

from "AmbulatoryVisits"

Graphical user interface, text, application

Description automatically generated