**SQL Hackathon January 2024 80 Queries**

**Team Number: Team1**

**Team Name: Schema Squad**

**Team Members: Kalai, Meenakshi 134, Meenakshi 130, Sharmila, Sudisha**

**Question 1:**

What % of the dataset had clinician reported records vs patient reported?

**Query:**

SELECT

report\_source,

COUNT(patient\_id) AS patient\_count,

ROUND(COUNT(patient\_id)\*100/ SUM(COUNT(patient\_id)) OVER(),2) ||'%' AS patient\_percentage

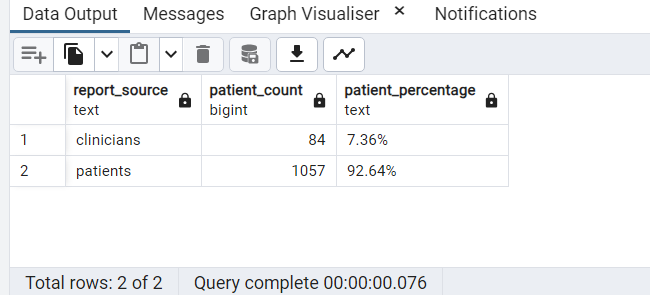
FROM

patient\_details

GROUP BY

Report\_source;

**Result:**

****

**Question 2:**

Concatenate Patient ID and all symptoms as meaningful words into one string

**Query:**

SELECT

patient\_id || ' - ' ||

CASE

WHEN covid19\_sympt\_chills = 0 AND covid19\_sympt\_dry\_cough = 0 AND covid19\_sympt\_fatigue = 0

AND covid19\_sympt\_fever = 0 AND covid19\_sympt\_loss\_smell\_taste = 0 AND covid19\_sympt\_nasal\_congestion = 0

AND covid19\_sympt\_pain = 0 AND covid19\_sympt\_pneumonia = 0 AND covid19\_sympt\_shortness\_breath = 0

AND covid19\_sympt\_sore\_throat = 0

THEN ' no symptoms'

ELSE

CONCAT(CASE WHEN covid19\_sympt\_chills = 1 THEN 'Chills, ' END,

CASE WHEN covid19\_sympt\_dry\_cough = 1 THEN 'Dry cough, ' END,

CASE WHEN covid19\_sympt\_fatigue = 1 THEN 'Fatigue, ' END,

CASE WHEN covid19\_sympt\_fever = 1 THEN 'Fever, ' END,

CASE WHEN covid19\_sympt\_loss\_smell\_taste = 1 THEN 'Loss of smell and taste, ' END,

CASE WHEN covid19\_sympt\_nasal\_congestion = 1 THEN 'Nasal Congestion, ' END,

CASE WHEN covid19\_sympt\_pain = 1 THEN 'Pain, ' END,

CASE WHEN covid19\_sympt\_pneumonia = 1 THEN 'Pneumonia, ' END,

CASE WHEN covid19\_sympt\_shortness\_breath = 1 THEN 'Shortness of breath, ' END,

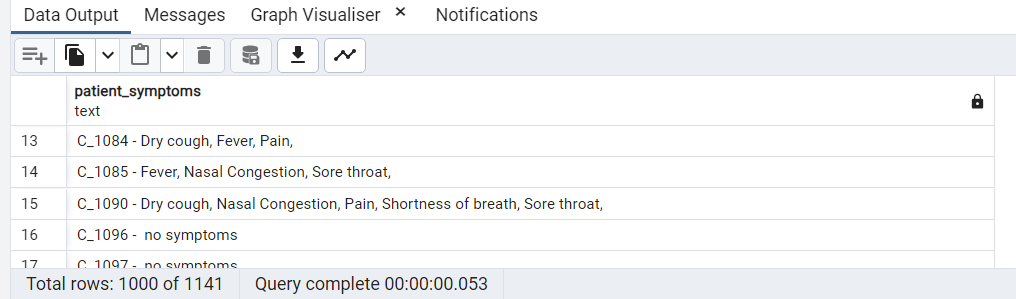
CASE WHEN covid19\_sympt\_sore\_throat = 1 THEN 'Sore throat,' END)

END as patient\_symptoms

FROM

Covid\_symptoms;

**Result:**

****

**Question 3:**

What percentage of those in the ICU are smokers?

**Query:**

SELECT

SUM(CASE WHEN cd.covid19\_icu\_stay = 1

AND pd.smoker = 1 THEN 1 ELSE 0 END)\*100/

NULLIF(SUM(CASE WHEN cd.covid19\_icu\_stay = 1 THEN 1 ELSE 0 END),0)

|| '%' AS percentage\_smoker\_icu

FROM

patient\_details pd

INNER JOIN

covid\_details cd

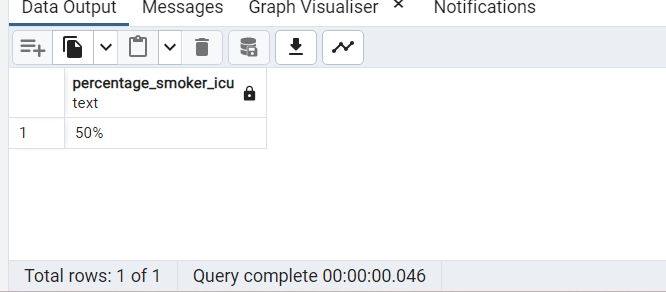
ON

pd.patient\_id = cd.patient\_id

WHERE

cd.covid19\_icu\_stay = 1;

**Result:**

****

**Question 4:**

Create a crosstab with each type of MS as columns and covid

--diagnosed/not diagnosed as rows and show count of patients in each group.

**Query:**

SELECT

ms\_type2,

COUNT(pd.patient\_id) AS totalpatients,

SUM(CASE WHEN cd.covid19\_confirmed\_case = 1 THEN 1 ELSE 0 END) AS "Diagnosed",

SUM(CASE WHEN cd.covid19\_confirmed\_case = 0 THEN 1 ELSE 0 END) AS "Not Diagnosed"

FROM

patient\_details pd

LEFT JOIN

covid\_details cd

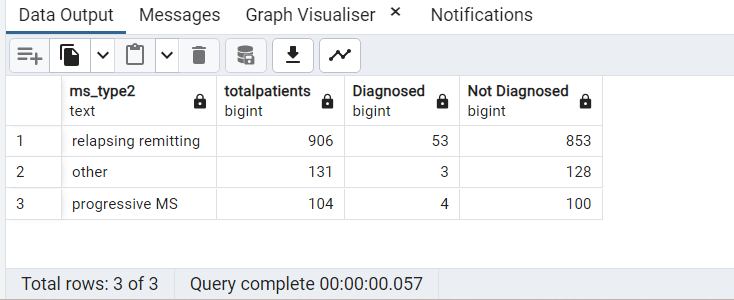
ON

pd.patient\_id = cd.patient\_id

GROUP BY

ms\_type2;

**Result:**

****

**Question 5:**

How many patients in the dataset are diagnosed with COVID and are on a ventilator?

**Query:**

--Solution 1:

SELECT

COUNT(patient\_id) AS patientscount\_covid\_ventilation

FROM

covid\_details

WHERE

covid19\_diagnosis = 'confirmed'

AND covid19\_ventilation =1;

-- Solution 2:

SELECT

COUNT(patient\_id) AS patientscount\_covid\_ventilation

FROM

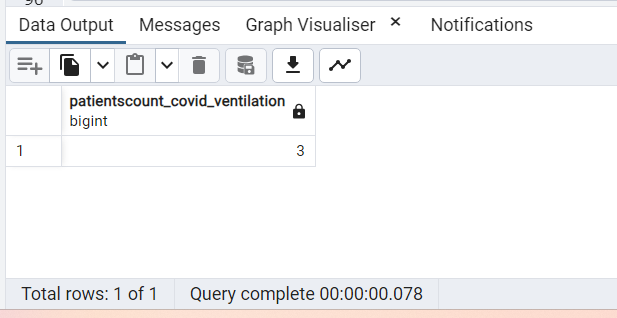
covid\_details

WHERE

covid19\_confirmed\_case = 1

AND covid19\_ventilation =1;

**Result:**

****

**Question 6:**

Using windows functions show month of onset and the number of hospitalizations in the next month

**Query:**

--Description: COALESCE function replaces null value to 0 (last row of the result )

--Assumption: zero is displayed ,if no of records are found for the next month

--ex.2nd row: for 2020-01 there are no records available for 2020-02 so zero is displayed

SELECT

to\_char(date\_of\_onset, 'YYYY-MM') AS onset\_month,

COALESCE (CASE WHEN

LEAD(concat(to\_char(date\_of\_onset, 'YYYY-MM'),'-01'),1)

OVER()::date != concat(to\_char(date\_of\_onset, 'YYYY-MM'),'-01')::date + interval '1 month'

THEN 0

ELSE LEAD(COUNT(patient\_id),1) OVER() END,0) as noOfHospitalizationInNextMonth

FROM

covid\_details

WHERE

covid19\_admission\_hospital=1

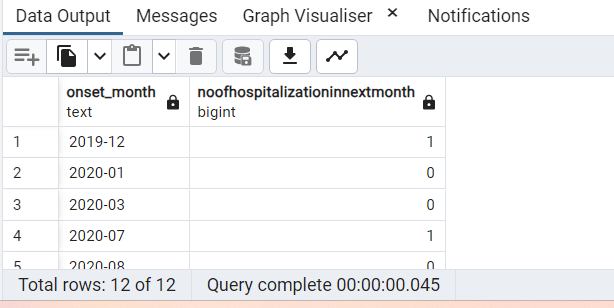
GROUP BY

to\_char(date\_of\_onset, 'YYYY-MM')

ORDER BY

to\_char(date\_of\_onset, 'YYYY-MM');

**Result:**

****

**Question 7:**

How many patients have covid and have reported symptoms of chills?

**Query:**

SELECT

COUNT(cs.patient\_id) AS num\_patients\_with\_covid\_chills

FROM

covid\_details cd

JOIN

covid\_symptoms cs

ON

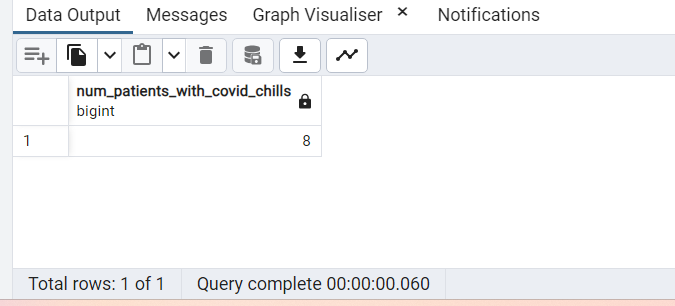
cs.patient\_id = cd.patient\_id

WHERE

cd.covid19\_confirmed\_case = 1

AND cs.covid19\_sympt\_chills = 1;

**Result:**

****

**Question 8:**

Add a column to the patient table which translates age\_in\_cat to the following values.

--(0: age range <18, 1: age range 18-50,2: age range 51-70,3: age range >71)

**Query:**

-- create a procedure to add column and update the column

CREATE OR REPLACE PROCEDURE catAge\_to\_HumanAge()

AS

$$

BEGIN

-- Add a new column to the table if it does not exists

ALTER TABLE patient\_details ADD COLUMN IF NOT EXISTS human\_age varchar(20);

-- Update the new column with the data

UPDATE patient\_details

SET human\_age = CASE

WHEN age\_in\_cat=0 THEN '<18'

WHEN age\_in\_cat=1 THEN '18-50'

WHEN age\_in\_cat=2 THEN '51-70'

WHEN age\_in\_cat=3 THEN '>71'

END;

END;

$$

LANGUAGE plpgsql;

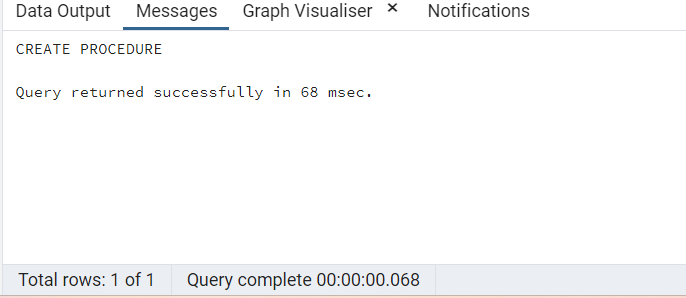
-- call the stored procedure

CALL catAge\_to\_HumanAge();

-- Select to see results

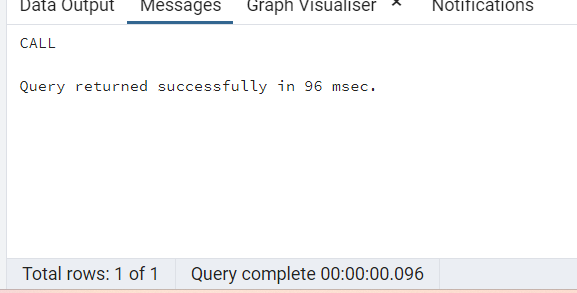
select \* from patient\_details;

**Result:**

****

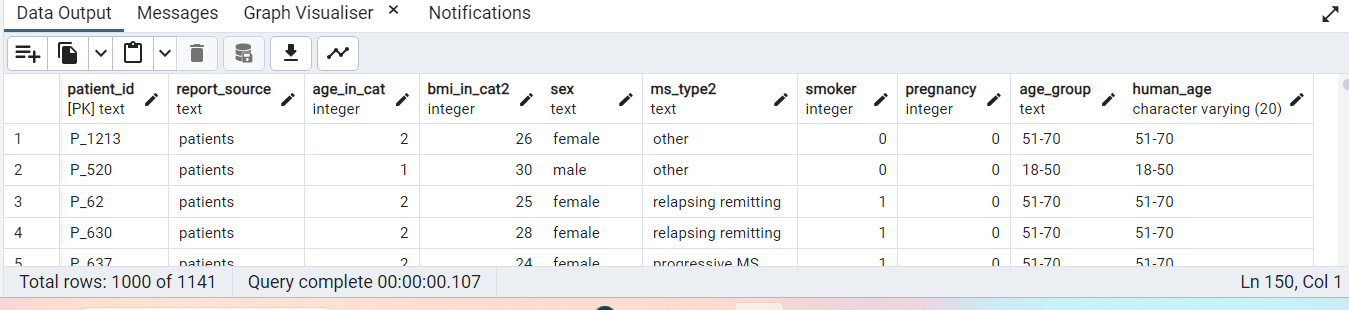
-- call the stored procedure

CALL catAge\_to\_HumanAge();

****

-- Select to see results

select \* from patient\_details;

****

**Question 9:**

Do this within a stored procedure"

How many patients who recovered had no comorbidities?

**Query:**

CREATE OR REPLACE FUNCTION public.get\_comorbidities\_sum(

patientid text)

RETURNS integer

LANGUAGE 'plpgsql'

COST 100

VOLATILE PARALLEL UNSAFE

AS $BODY$

DECLARE

sum\_comorbidities integer;

BEGIN

SELECT SUM(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease + com\_diabetes +

com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy + com\_neurological\_neuromuscular)

INTO sum\_comorbidities

FROM comorbidities

WHERE patient\_ID = patientID;

RETURN sum\_comorbidities;

END;

$BODY$;

-- Solution Query

WITH recovered\_no\_comorbidities

AS

(

SELECT

pd.patient\_id,

get\_comorbidities\_sum(pd.patient\_id) AS comorbidities\_sum

FROM

patient\_details pd

JOIN

covid\_details cd

ON

pd.patient\_id = cd.patient\_id

WHERE

covid19\_outcome\_recovered = 1

)

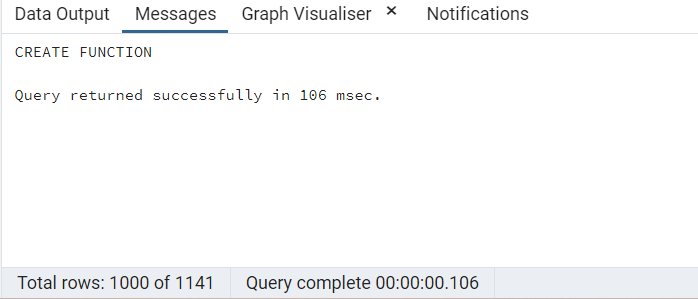
SELECT

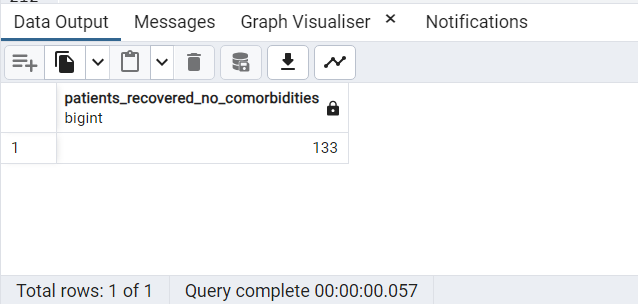
COUNT(patient\_id) as patients\_recovered\_no\_comorbidities

FROM recovered\_no\_comorbidities

WHERE comorbidities\_sum = 0;

**Result:**

****

****

**Question 10:**

What % of overweight patients have COVID?

**Query:**

WITH overweight\_covid AS

(

SELECT

cd.covid19\_confirmed\_case,

to\_char((count(pd.patient\_id) \* 100.0 / sum(count(pd.patient\_id)) OVER ()), 'FM90.0" %"') AS Percentage

FROM

patient\_details pd

JOIN

covid\_details cd

ON

pd.patient\_id = cd.patient\_id

WHERE

pd.bmi\_in\_cat2 >= 25

GROUP BY

cd.covid19\_confirmed\_case

)

SELECT

percentage as percentage\_overweight\_covid

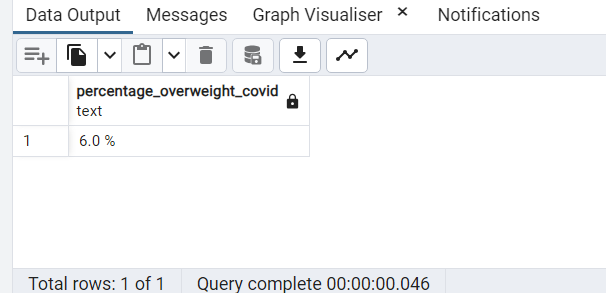
FROM

overweight\_covid

WHERE

covid19\_confirmed\_case = 1;

**Result:**

****

**Question 11:**

What is the correlation between smoking and being on a ventilator?

**Query:**

SELECT

ROUND(CORR(pd.smoker, cd.covid19\_ventilation):: numeric,3) AS Correlation

FROM

patient\_details pd

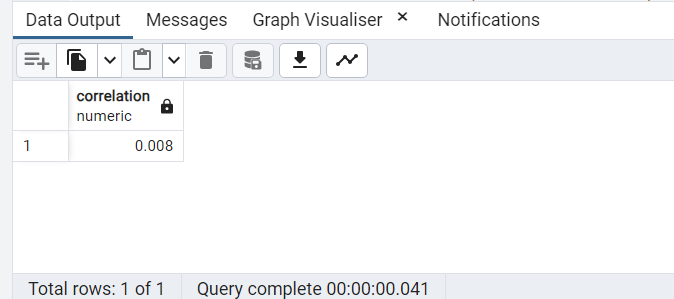
JOIN

covid\_details cd

ON

pd.patient\_id=cd.patient\_id;

**Result:**

****

**Question 12:**

Display an array of personal markers:BMI, EDSS, MS Type for every patient. The result should look like this

**Query:**

SELECT

pd.patient\_id,

ARRAY[pd.ms\_type2, pd.bmi\_in\_cat2, pm.edss\_in\_cat2 ]::text[] AS mstype\_bmi\_edss\_marker

FROM

patient\_details pd

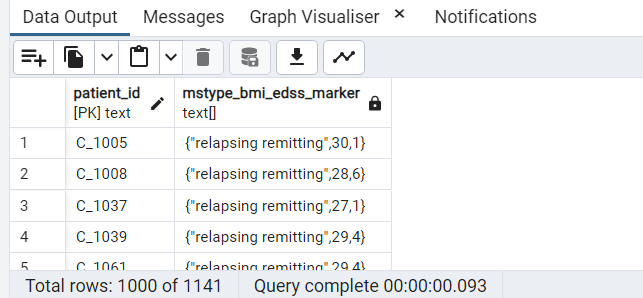
JOIN

patient\_msdetails pm

ON

pd.patient\_id = pm.patient\_id;

**Result:**

****

**Question 13:**

What is the Standard deviation, mean and variance from mean in EDSS levels for all patients.

**Query:**

SELECT

STDDEV(edss\_in\_cat2) AS Standard\_dev,

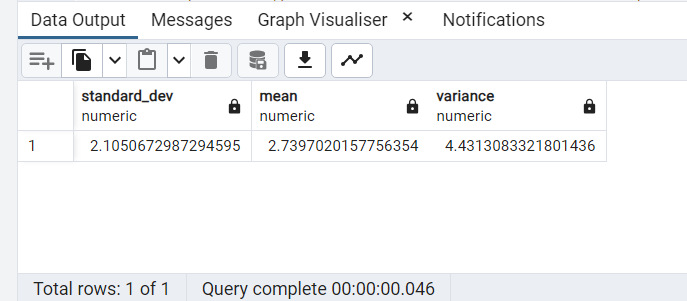
AVG(edss\_in\_cat2) as Mean,

VARIANCE(edss\_in\_cat2) as Variance

FROM

patient\_msdetails;

**Result:**

****

**Question 14:**

Create a materialized view with patient details (for all those with covid) and

assign a COVID severity score to each patient assigning 1 point each for these symptoms.

(cardiovascular\_disease,chronic\_kidney\_disease,chronic\_liver\_disease,diabetes,hypertension,

immunodeficiency,lung\_disease,malignancy,neurological\_neuromuscular)

**Query:**

-- DROP MATERIALIZED VIEW IF EXISTS;

DROP MATERIALIZED VIEW view\_covidSeverity;

-- Creation of Materialized View

CREATE MATERIALIZED VIEW view\_covidSeverity

AS

SELECT cd.patient\_id,pd.report\_source category,pd.sex,pd.age\_group,

(SELECT SUM(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease + com\_diabetes +

com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy + com\_neurological\_neuromuscular) AS covid\_severity

FROM comorbidities como WHERE cd.patient\_id = como.patient\_id)

FROM covid\_details cd JOIN comorbidities com

ON cd.patient\_id = com.patient\_id

JOIN patient\_details pd

ON cd.patient\_id = pd.patient\_id

WHERE covid19\_confirmed\_case = 1

ORDER BY covid\_severity DESC

WITH NO DATA;

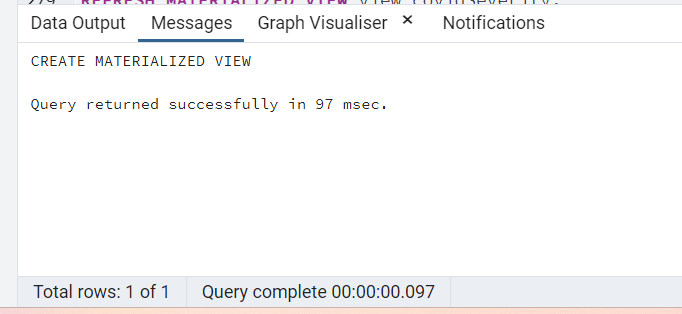
-- Refresh the view to load the data

REFRESH MATERIALIZED VIEW view\_covidSeverity;

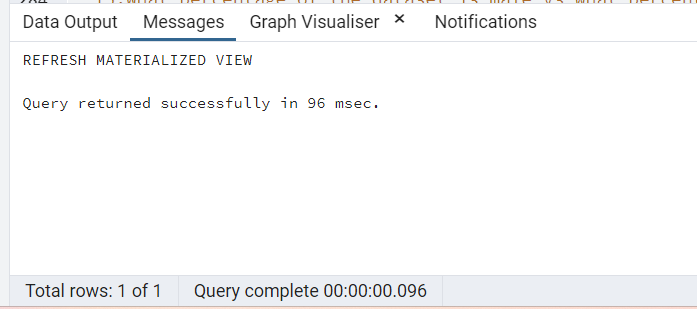
-- Select to see the results

SELECT \* FROM view\_covidSeverity;

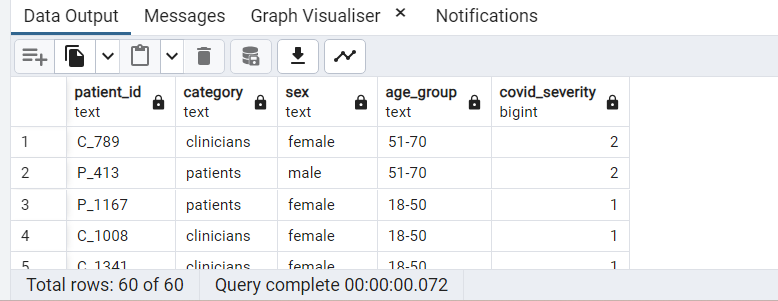
**Result:**

****

-- Refresh the view to load the data

****

-- Select to see the results

****

**Question 15:**

What percentage of the dataset is male vs what percentage is female? Calculate without the Use of a subquery or CTE

**Query:**

-- Solution 1

SELECT

ROUND((COUNT(CASE WHEN sex = 'male' THEN 1 END) \* 100.0 / COUNT(\*)),2) || '%' AS percent\_male,

ROUND((COUNT(CASE WHEN sex = 'female' THEN 1 END) \* 100.0 / COUNT(\*)),2) ||'%' AS percent\_female

FROM

patient\_details;

-- Solution 2

SELECT

sex, count(patient\_id),

to\_char((count(patient\_id) \* 100.0 / sum(count(patient\_id)) OVER ()), 'FM90.00" %"') AS Percentage

FROM

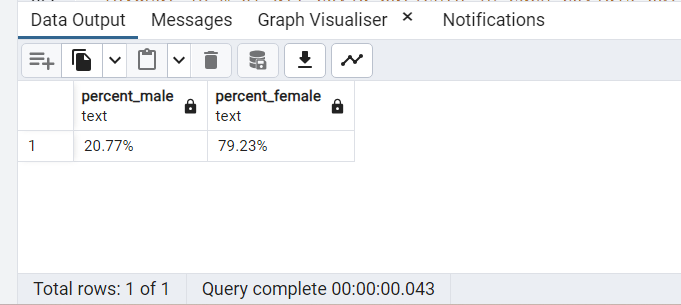
patient\_details

GROUP BY

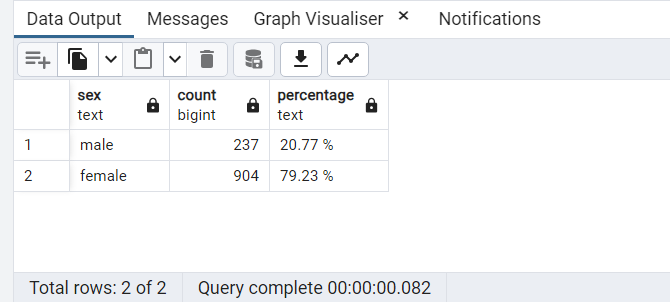
Sex;

**Result:**

-- Solution 1

****

-- Solution 2

****

**Question 16:**

Create a function that checks if the patientId entered is a smoker. It should return a true/false answer.

**Query:**

CREATE OR REPLACE FUNCTION patient\_issmoker(patientID varchar)

RETURNS boolean

AS

$body$

BEGIN

IF EXISTS(

SELECT patient\_id

FROM patient\_details

WHERE patient\_id = patientID AND smoker = 1

)THEN

RETURN true;

ELSE

RETURN false;

END IF;

END;

$body$ LANGUAGE plpgsql;

--Verify to view the results --

SELECT patient\_issmoker('C\_1091') AS smoker; -- returns true

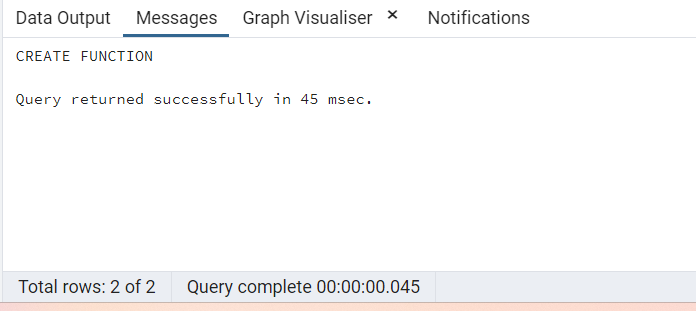
SELECT patient\_issmoker('P\_123') AS smoker; -- returns false

--verify the output ---

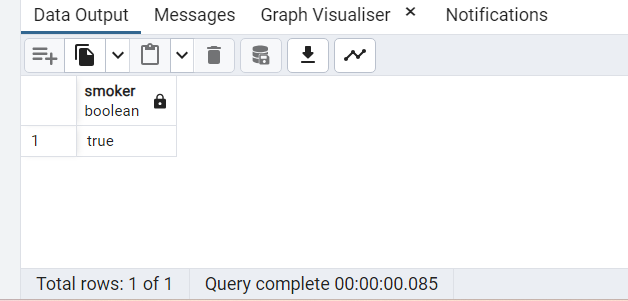
SELECT \* FROM patient\_details WHERE patient\_id='C\_1091';

SELECT \* FROM patient\_details WHERE patient\_id='P\_123';

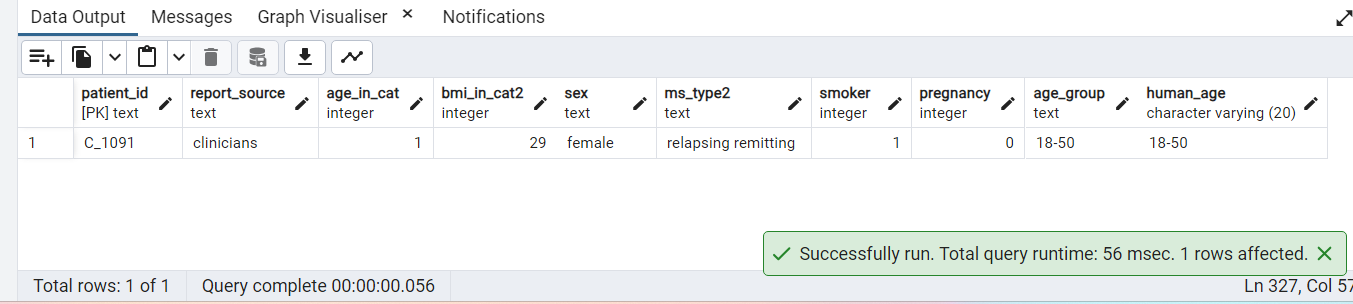
**Result:**

****

--Verify to view the results --

****

--verify the output ---

****

**Question 17:**

How many patients are prescribed drugs have names starting with 'd'

**Query:**

SELECT

COUNT(patient\_id) as patientsCount

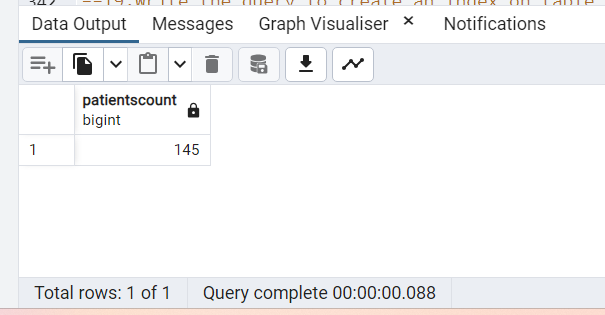
FROM

patient\_msdetails

WHERE

prescribed\_drug like 'd%'

**Result:**

****

**Question 18:**

What is % of all covid patients in each covid19 outcome levels? Use a windows function without a subquery.

**Query:**

SELECT

covid19\_outcome\_levels\_2,

COUNT(patient\_id) as No\_of\_Patients,

to\_char((count(patient\_id) \* 100.0 / sum(count(patient\_id)) OVER ()), 'FM90.0" %"') AS Percentage

FROM

covid\_details

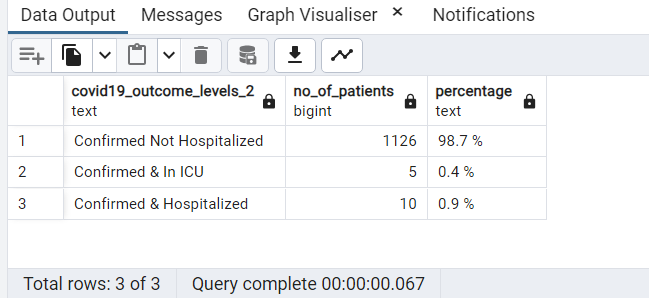
GROUP BY

covid19\_outcome\_levels\_2

ORDER BY

covid19\_outcome\_levels\_2 DESC;

**Result:**

****

**Question 19:**

Write the query to create an Index on table public.comorbidities by

--selecting a column. Show the index using a query and write a query drop the same index.

**Query:**

-- Query to verify before and after creating index

EXPLAIN ANALYZE

SELECT

pd.patient\_id,pd.sex,co.com\_hypertension

FROM

Patient\_details pd

JOIN

comorbidities co

ON pd.patient\_id = co.patient\_id

WHERE

sex= 'female' AND co.com\_hypertension=1

-- Create index query

CREATE INDEX index\_hypertension

ON comorbidities (com\_hypertension);

-- Check if index is created

SELECT

indexname,

indexdef

FROM

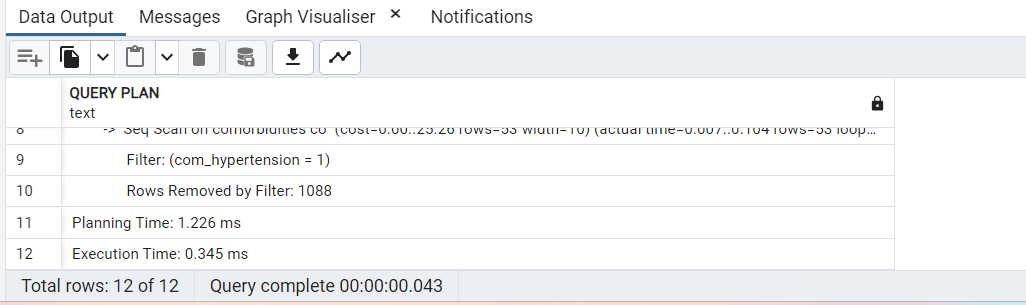
pg\_indexes

WHERE

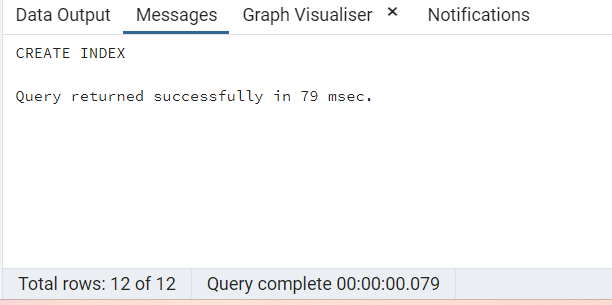
tablename = 'comorbidities';

**Result:**

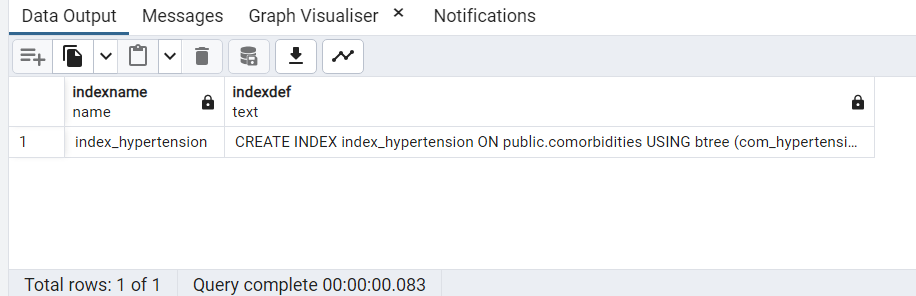
-- Query to verify before and after creating index

****

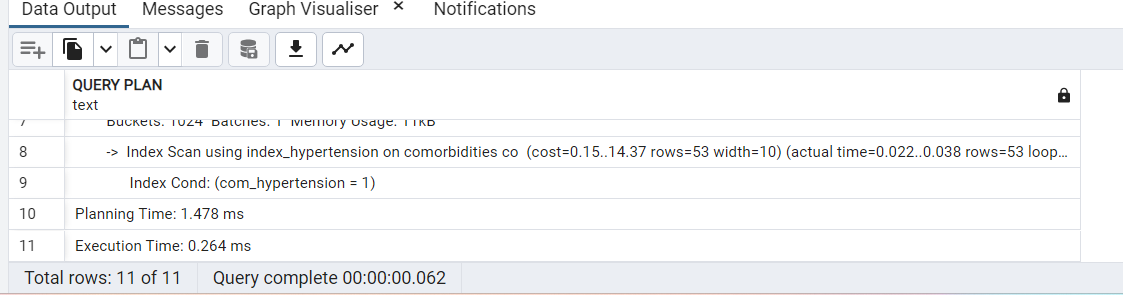
-- Create index query

****

-- Check if index is created

****

-- Query to verify before and after creating index

****

**Question 20:**

What % of total symptoms is Fatigue?

**Query:**

SELECT

covid19\_sympt\_fatigue,

CASE

WHEN covid19\_sympt\_fatigue=1 THEN 'Has Fatigue Symptom'

WHEN covid19\_sympt\_fatigue=0 THEN 'No Fatigue Symptom'

END AS FatigueSymptom,

COUNT(patient\_id) as NoofPatients,

to\_char((count(patient\_id) \* 100.0 / sum(count(patient\_id)) OVER ()), 'FM90.0" %"') AS Percentage

FROM

covid\_symptoms

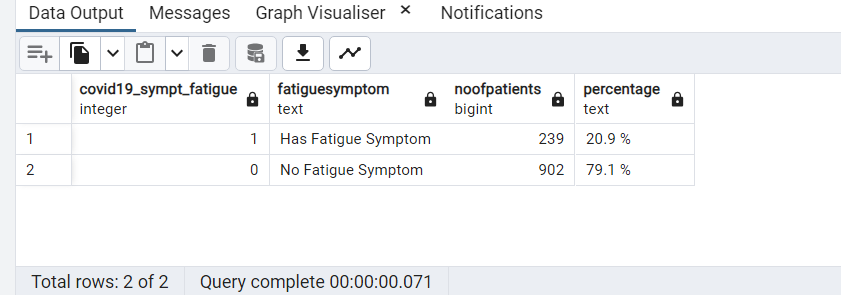
GROUP BY

covid19\_sympt\_fatigue

ORDER BY

covid19\_sympt\_fatigue DESC;

**Result:**

****

**Question 21:**

List all patients that were admitted between 2020 January and 2020 March

**Query:**

-- Solution 1:

SELECT

\*

FROM

public.covid\_details

WHERE

date\_of\_onset BETWEEN '2020-01-01' AND '2020-03-31'

AND covid19\_admission\_hospital = 1

ORDER BY

patient\_id;

-- Solution 2:

SELECT

\*

FROM

public.covid\_details

WHERE

DATE\_PART('MONTH', date\_of\_onset) IN (1,2,3) AND DATE\_PART('YEAR', date\_of\_onset) = 2020

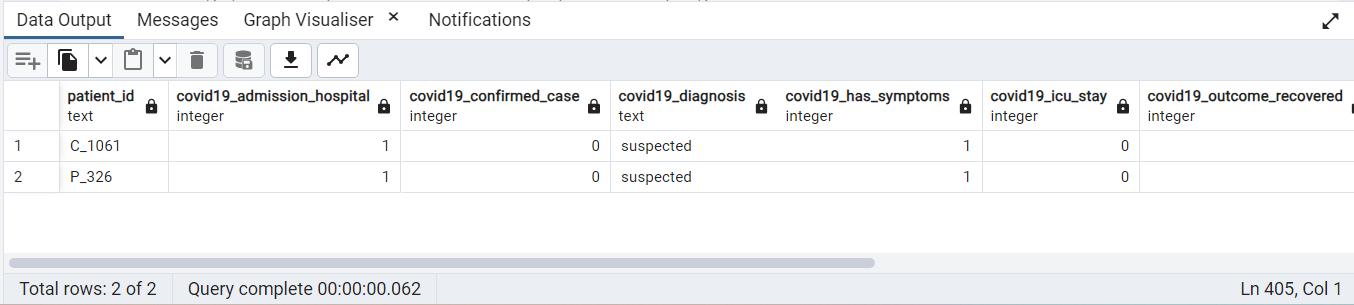
AND covid19\_admission\_hospital = 1

ORDER BY

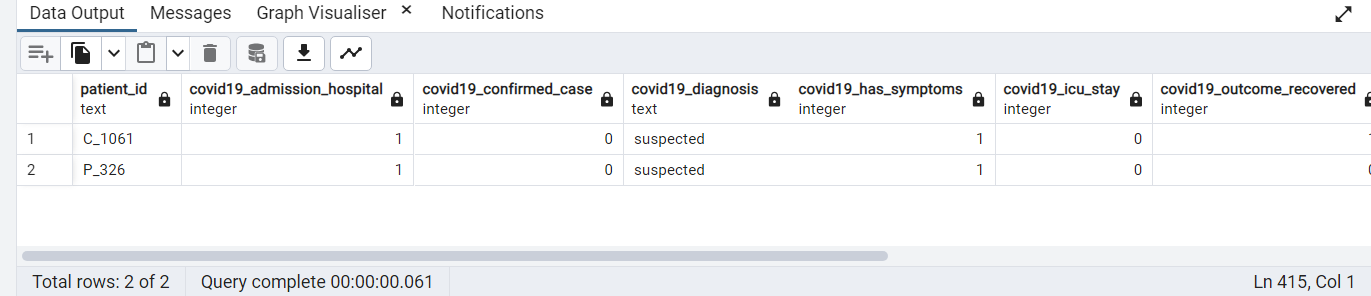
patient\_id;

**Result:**

-- Solution 1:

****

-- Solution 2:



**Question 22:**

On average, how many comorbidities do people with 'progressive\_MS' have

**Query:**

SELECT ROUND(AVG(comorbidities\_SUM),2) AS Avg\_comorbidities

FROM

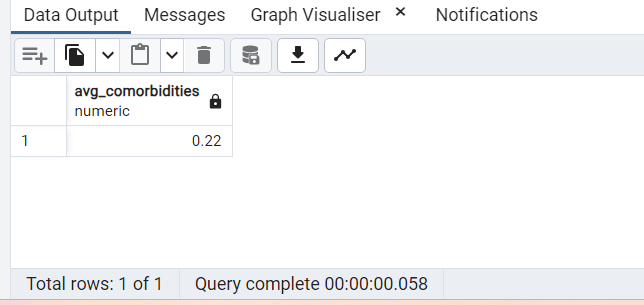
(SELECT patient\_id,sum(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease + com\_diabetes +

com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy + com\_neurological\_neuromuscular) AS comorbidities\_SUM

FROM comorbidities WHERE patient\_id IN (SELECT patient\_id FROM patient\_details WHERE ms\_type2='progressive MS')

GROUP BY patient\_id)

**Result:**

****

**Question 23:**

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

**Query:**

-- Drop the view if it exists

DROP View IF EXISTS view\_NoSchemaTable;

-- Create or replace view with the name "view\_NoSchemaTable"

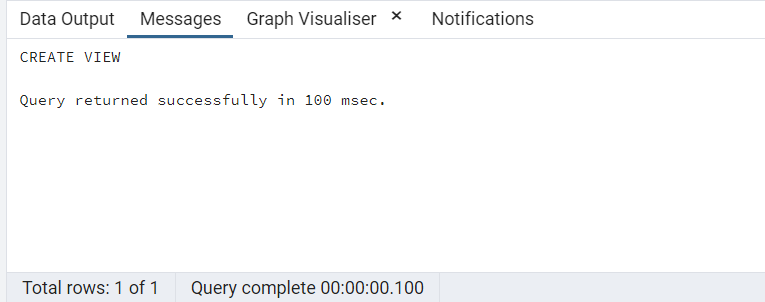
CREATE OR REPLACE VIEW view\_NoSchemaTable AS SELECT 'View created without Schema or Table' AS Information;

-- Check the view using select statement

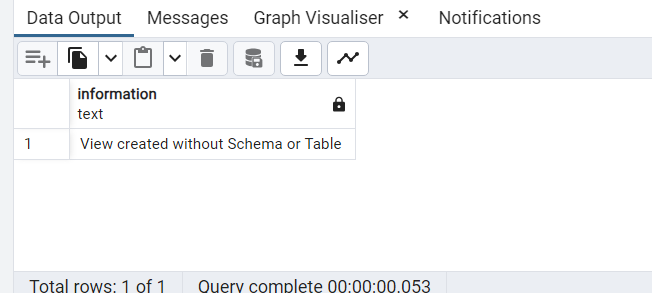
SELECT \* FROM view\_NoSchemaTable;

**Result:**

-- Create or replace view with the name "view\_NoSchemaTable"

****

-- Check the view using select statement

****

**Question 24:**

Create a crosstab with each type of MS as columns and the medications used as rows

**Query:**

SELECT \*

FROM crosstab(

'SELECT pd.patient\_id,pd.ms\_type2,pm.prescribed\_drug

FROM Patient\_details pd

JOIN patient\_msdetails pm

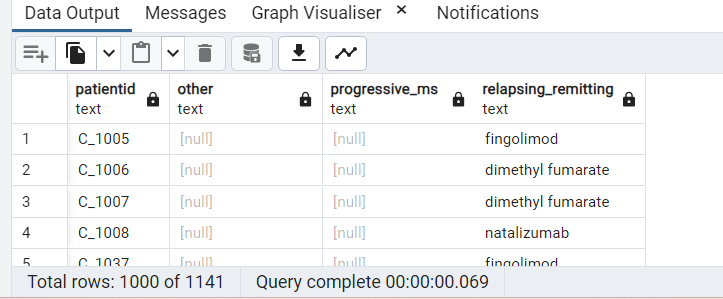
ON pd.patient\_id = pm.patient\_id

ORDER BY 1,2',

'SELECT distinct ms\_type2 from Patient\_details order by 1'

)AS crosstab\_mstype (PatientID text,other text, progressive\_MS text, relapsing\_remitting text);

**Result:**

****

**Question 25:**

"Get the patient's ID who has a max severity score from the materialized view created in Q14 using windows functions.

**Query:**

SELECT

patient\_id,covid\_severity

FROM

(SELECT patient\_id,covid\_severity,

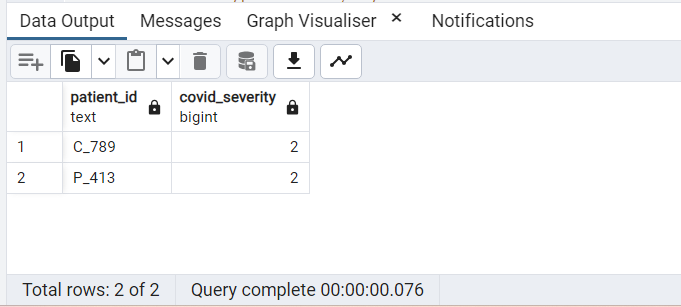
RANK() OVER (ORDER BY covid\_severity DESC) AS rownum

FROM view\_covidSeverity)

WHERE

rownum = 1;

**Result:**

****

**Question 26:**

Rank the types of MS, by the sum of all comorbidities associated with each one.

**Query:**

-- Considering the highest number as Rank 1 here

-- Solution 1

WITH MS\_comorbidities AS

(

SELECT

pd.ms\_type2,

(SELECT sum(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease +

com\_diabetes + com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy +

com\_neurological\_neuromuscular) AS comorbidities\_SUM

FROM comorbidities como

WHERE pd.patient\_id = como.patient\_id)

FROM

patient\_details pd

JOIN

comorbidities como

ON

pd.patient\_id = como.patient\_id

)

SELECT

ms\_type2,SUM(comorbidities\_SUM) as comorbiditiesSUM,

RANK () OVER (ORDER BY SUM(comorbidities\_SUM) DESC) comorbidities\_rank

FROM

MS\_comorbidities

GROUP BY

ms\_type2;

-- Solution 2:

SELECT

pd.ms\_type2,

sum(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease +

com\_diabetes + com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy +

com\_neurological\_neuromuscular) AS comorbidities\_SUM,

RANK () OVER (ORDER BY sum(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease +

com\_diabetes + com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy +

com\_neurological\_neuromuscular) DESC) comorbidities\_rank

FROM

patient\_details pd

JOIN

comorbidities como

ON

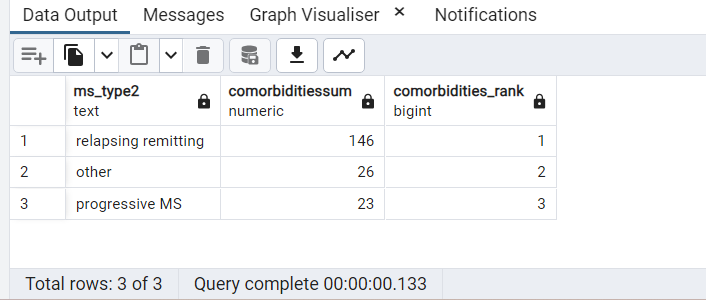
pd.patient\_id = como.patient\_id

GROUP BY

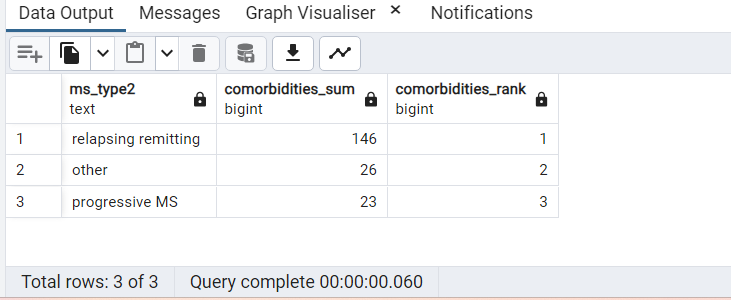
pd.ms\_type2;

**Result:**

-- Solution 1



-- Solution 2



**Question 27:**

What % of patients in the ICU needed ventilation. Calculate and without a subquery or CTE.

**Query:**

SELECT

covid19\_ventilation,

CASE

WHEN covid19\_ventilation = 0 THEN 'Ventilation Not Required'

WHEN covid19\_ventilation = 1 THEN 'Ventilation Required'

END,

count(patient\_id) AS No\_of\_Patients,

to\_char((count(patient\_id) \* 100.0 / sum(count(patient\_id)) OVER ()), 'FM90.0" %"') AS Percentage

FROM

covid\_details

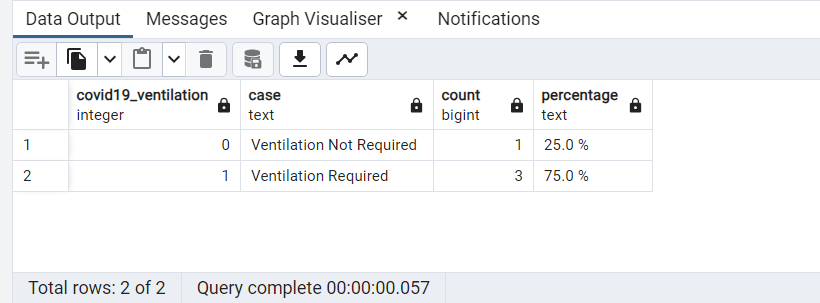
WHERE

covid19\_icu\_stay = 1

GROUP BY

covid19\_ventilation;

**Result:**

****

**Question 28:**

Using mean and std\_dev create your own 10 Patient\_ids and EDSS values that have the same std\_dev as the original table

**Query:**

**Result:**

**Question 29:**

calculate the frequency of EDSS and cumulative frequency of EDSS of the patients in the dataset

**Query:**

SELECT

edss\_in\_cat2 AS EDSS\_Value,

COUNT(patient\_id) AS EDSS\_Frequency,

SUM(COUNT(patient\_id)) OVER (ORDER BY edss\_in\_cat2) AS EDSS\_Cumulative

FROM

patient\_msdetails

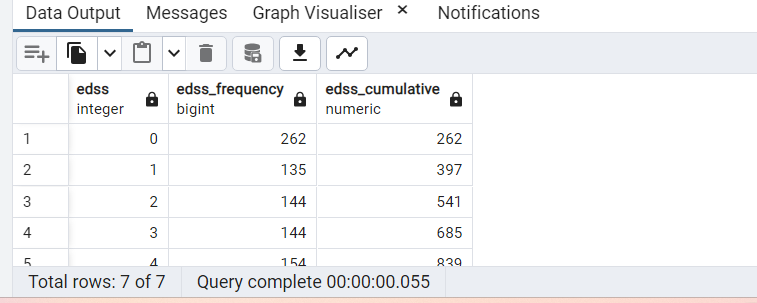
GROUP BY

edss\_in\_cat2

ORDER BY

edss\_in\_cat2;

**Result:**

****

**Question 30:**

How many people who have covid are overweight vs those who are not overweight?

**Query:**

--Considering the universal bmi range and have taken value 25

SELECT

CASE

WHEN pd.bmi\_in\_cat2 >= 25 THEN 'Overweight'

WHEN pd.bmi\_in\_cat2 < 25 THEN 'Not Overweight'

END AS BMICategory,

COUNT(pd.bmi\_in\_cat2) as patientCount

FROM

patient\_details pd

JOIN

covid\_details cd

ON

pd.patient\_id = cd.patient\_id

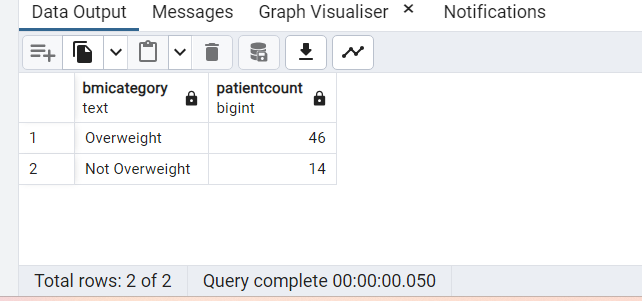
WHERE

covid19\_diagnosis = 'confirmed'

GROUP BY

BMICategory;

**Result:**

****

**Question 31:**

What percentage of each MStype are smokers?

**Query:**

SELECT

ms\_type2 AS MSType,

count(patient\_id) as PatientCount,

to\_char((count(patient\_id) \* 100.0 / sum(count(patient\_id)) OVER ()), 'FM90.0" %"') AS Percentage

FROM

patient\_details

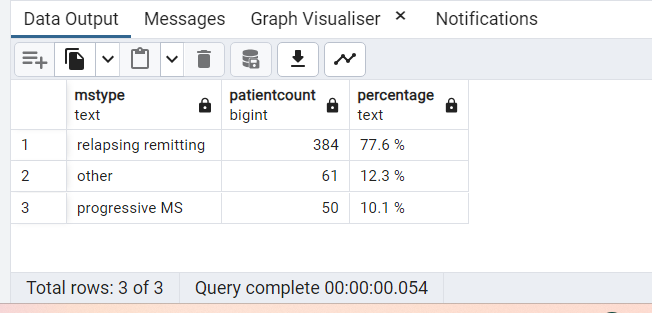
WHERE

smoker = 1

GROUP BY

ms\_type2;

**Result:**

****

**Question 32:**

Divide avg BMI by avg EDSS and return number of decimals in the result

**Query:**

SELECT

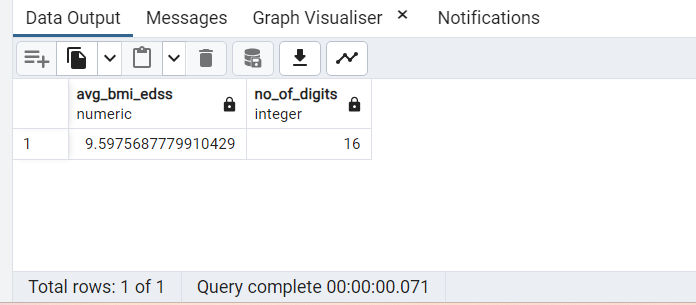
(SELECT AVG(bmi\_in\_cat2) AS Avg\_bmi FROM patient\_details) /

(SELECT AVG(edss\_in\_cat2) AS Avg\_edss FROM patient\_msdetails) AS Avg\_BMI\_EDSS,

SCALE((SELECT AVG(bmi\_in\_cat2) AS Avg\_bmi FROM patient\_details) /

(SELECT AVG(edss\_in\_cat2) AS Avg\_edss FROM patient\_msdetails)) AS No\_of\_digits;

**Result:**

****

**Question 33:**

List all Patients who are in the youngest age group and did not recover from COVID

**Query:**

--Assumption: not recovered=0, recovered=1

SELECT

patient\_id

FROM

patient\_details

JOIN

covid\_details covid USING (patient\_id)

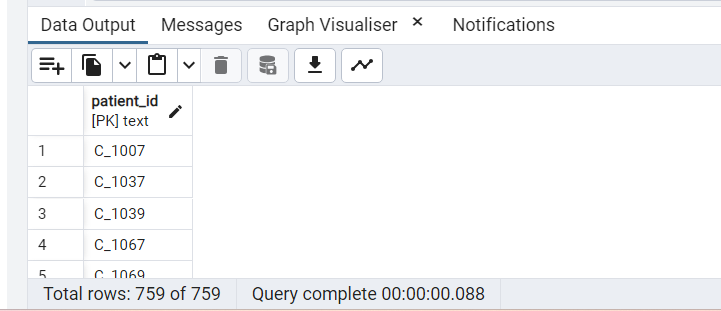
WHERE

covid19\_outcome\_recovered=0 AND age\_in\_cat =1

ORDER BY

patient\_id;

**Result:**

****

**Question 34:**

Extract the week from the Date Of Onset and show the distribution of week numbers using width bucket function.

**Query:**

-- Description: Created temporary table ‘weeks’ to calculate ‘maxNoOfWeeks’ :

-- To Avoid hard coding

-- To Avoid running subquery in the main query for two times to calculate max no of weeks.

WITH weeks AS (

SELECT MAX(DATE\_PART('week', date\_of\_onset))::int maxNoOfWeeks

FROM covid\_details

) --created maxNoOfWeeks

SELECT

WIDTH\_BUCKET(DATE\_PART('week', date\_of\_onset), 1,

(SELECT maxNoOfWeeks FROM weeks),

(SELECT maxNoOfWeeks FROM weeks)) AS week\_bucket,

COUNT(patient\_id) AS No\_of\_Patients

FROM

covid\_details

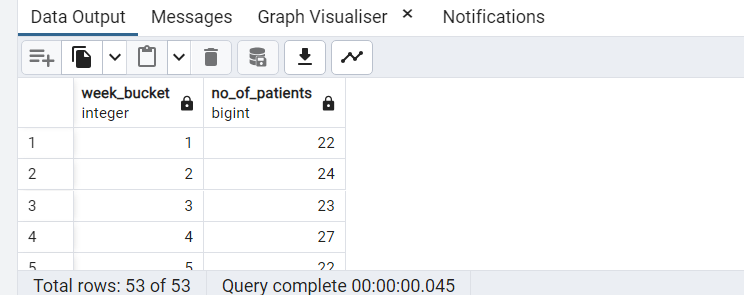
GROUP BY

week\_bucket

ORDER BY

week\_bucket;

**Result:**

****

**Question 35:**

How many patients were admitted in the current month of any month

**Query:**

-- Solution 1:

-- Create a function with month\_onset as input parameter

CREATE OR REPLACE FUNCTION getPatientCountBasedonMonth(

month\_onset int

)

-- returns patient count

RETURNS int

AS $$

DECLARE

No\_of\_Patients int;

BEGIN

SELECT COUNT(\*) INTO No\_of\_Patients FROM covid\_details WHERE

covid19\_admission\_hospital =1 AND EXTRACT('month' from date\_of\_onset) = month\_onset;

RETURN No\_of\_Patients;

END;

$$ LANGUAGE plpgsql;

-- execute the select statements to view the results.

SELECT getPatientCountBasedonMonth(1);

SELECT getPatientCountBasedonMonth(EXTRACT('month' FROM CURRENT\_DATE)::int);

-- Solution 2:

-- Create a procedure with month\_onset as input parameter

CREATE OR REPLACE PROCEDURE pr\_get\_noOFpaitent\_admitted\_in\_month(monthNo integer)

LANGUAGE plpgsql as

$$

DECLARE

BEGIN

DROP TABLE IF EXISTS MONTH\_TEMP;

CREATE TABLE month\_temp AS

SELECT TO\_CHAR(date\_of\_onset, 'Month') as month

,count(\*) as Patient\_Admitted

FROM covid\_details

WHERE covid19\_admission\_hospital =1

AND extract('month' from date\_of\_onset) = monthNo::integer

GROUP BY TO\_CHAR(date\_of\_onset, 'Month');

IF EXISTS (SELECT \* FROM MONTH\_TEMP) THEN

RAISE NOTICE using message ='PATIENTS GOT ADMITTED IN THIS MONTH

'||chr(10)||'PLEASE CHECK MONTH\_TEMP TABLE, USE'||CHR(10)||

'SELECT \* FROM MONTH\_TEMP';

ELSE

RAISE NOTICE 'No PATIENTS ADMITTED IN THIS MONTH';

END IF;

EXCEPTION

WHEN OTHERS THEN

RAISE NOTICE 'An error occured: %', SQLERRM;

END; $$

-- execute to see results of currect month of any month

CALL pr\_get\_noOFpaitent\_admitted\_in\_month(EXTRACT('month' FROM CURRENT\_DATE)::int);

-- execute to see result

CALL pr\_get\_noOFpaitent\_admitted\_in\_month(9); ---not found msg will get displayed

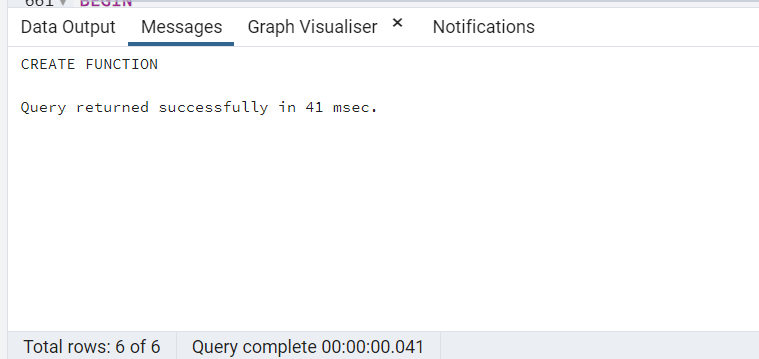
CALL pr\_get\_noOFpaitent\_admitted\_in\_month(1);

SELECT \* FROM month\_temp;

**Result:**

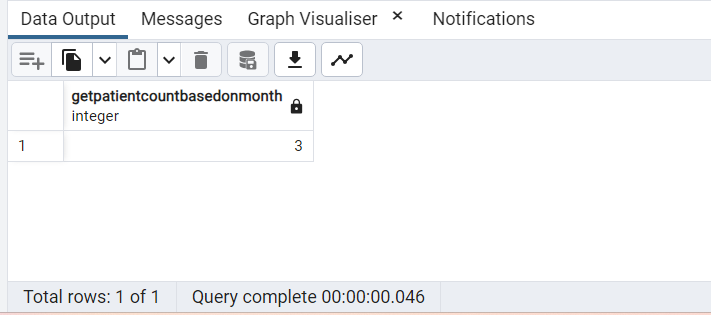
-- Solution 1:

-- Create a function with month\_onset as input parameter

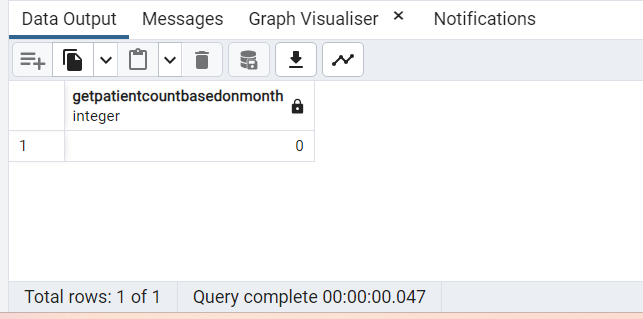
****

-- execute the select statements to view the results.

SELECT getPatientCountBasedonMonth(1);

****

SELECT getPatientCountBasedonMonth(EXTRACT('month' FROM CURRENT\_DATE)::int);



**Question 36:**

Which drug was most used for severe MS. Your answer should consider both frequency and duration of usage of drugs.

**Query:**

--Assumption:severeMS = ‘progressive MS’

SELECT

drug AS drug\_most\_used\_for\_severe\_MS

FROM (SELECT prescribed\_drug,

Row\_number() OVER (ORDER BY

COUNT(prescribed\_drug),MAX(duration\_treatment\_cat2)DESC) AS rowNo

FROM patient\_details

JOIN patient\_msdetails USING (patient\_id)

WHERE ms\_type2= 'progressive MS'

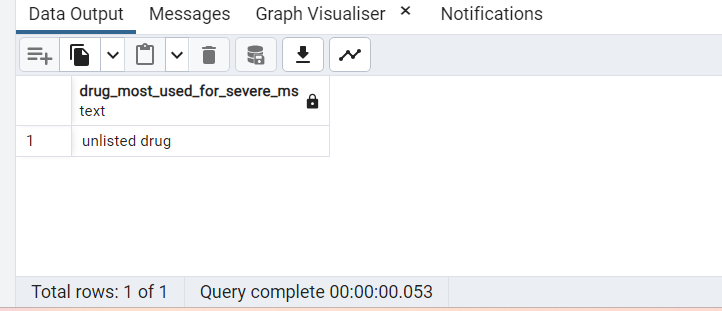
GROUP BY prescribed\_drug ,duration\_treatment\_cat2

ORDER BY COUNT(prescribed\_drug),duration\_treatment\_cat2 DESC

) AS DATA(drug,rowNo)

WHERE rowNo=1;

**Result:**

****

**Question 37:**

How many different types of medication are there in the dataset? Display count of patients against each one.

**Query:**

SELECT

prescribed\_drug,

COUNT(patient\_id) AS Count\_Of\_Patients

FROM

patient\_msdetails

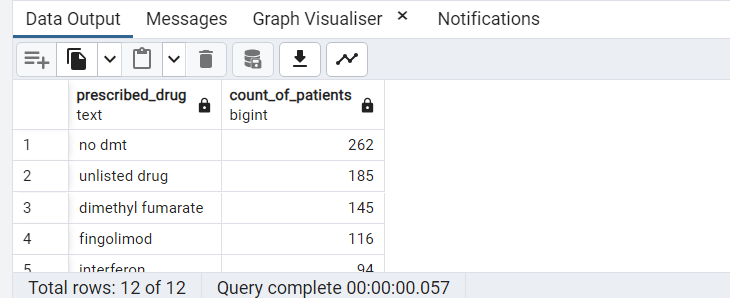
GROUP BY

prescribed\_drug

ORDER BY

COUNT(patient\_id) DESC;

**Result:**

****

**Question 38:**

Using windows functions show month of onset and the number of hospitalizations in the previous month

**Query:**

--Description: COALESCE function replaces null value to 0 (1st row of the result )

--Assumption: zero is displayed ,if no of records are not found in the previous month

--ex.3rd row: for 2020-03 there are no records available for 2020-02 so zero is displayed

SELECT

to\_char(date\_of\_onset, 'YYYY-MM') AS onset\_month,

COALESCE (CASE WHEN

LAG(concat(to\_char(date\_of\_onset, 'YYYY-MM'),'-01'),1)

OVER()::date != concat(to\_char(date\_of\_onset, 'YYYY-MM'),'-01')::date - interval '1 month'

THEN 0

ELSE

LAG(COUNT(patient\_id),1) OVER() END,0) AS Hospitalization\_PreviousMonth

FROM

covid\_details

WHERE

covid19\_admission\_hospital=1

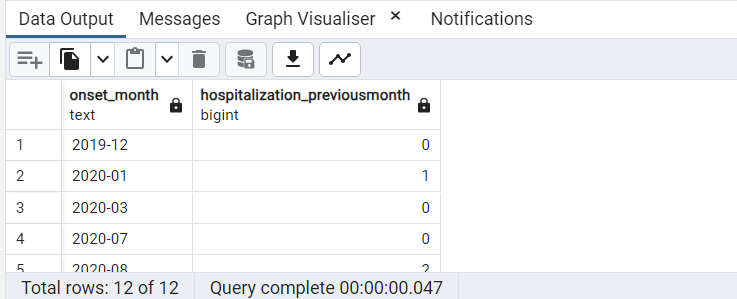
GROUP BY

to\_char(date\_of\_onset, 'YYYY-MM')

ORDER BY

to\_char(date\_of\_onset, 'YYYY-MM');

**Result:**

****

**Question 39:**

List all patients that were admitted during the busiest month in this dataset

**Query:**

-- ASSUMPTION: busiest month = The month and year having maximum records

--in the covid\_details table. '2020-05' having maximum records (47) of the dataset.

SELECT

patient\_id

FROM

covid\_details

WHERE

covid19\_admission\_hospital= 1

AND to\_char(date\_of\_onset, 'YYYY-MM') IN

(SELECT busiest\_month.monthYear FROM (

SELECT COUNT(patient\_id) AS cnt,

to\_char(date\_of\_onset, 'YYYY-MM') as monthYear,

RANK() OVER(ORDER BY COUNT(patient\_id) DESC) AS rnk

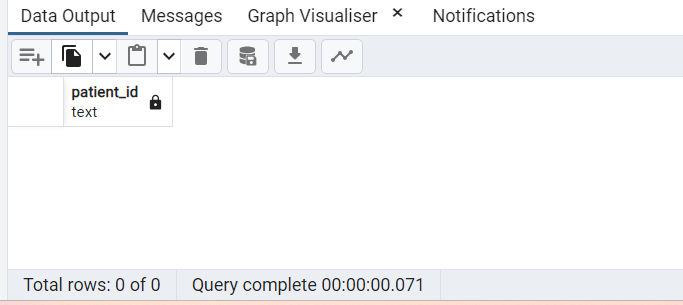
FROM covid\_details

GROUP BY to\_char(date\_of\_onset, 'YYYY-MM')

ORDER BY rnk) busiest\_month

WHERE rnk =1);

**Result:**

****

**Question 40:**

Create a function to calculate the total number of people who recovered for every drug entered.

**Query:**

-- Solution 1: Without cursor

CREATE OR REPLACE FUNCTION fn\_patientsRecoveredByDrug(drug text)

RETURNS TABLE(no\_Of\_Patient\_Recoverd bigint)

AS $$

BEGIN

RETURN QUERY

SELECT COUNT(c.patient\_id) as no\_Of\_Patient\_Recovered

FROM covid\_details c

RIGHT JOIN patient\_msdetails m using(patient\_id)

WHERE covid19\_outcome\_recovered=1

AND prescribed\_drug = drug

GROUP BY prescribed\_drug;

EXCEPTION

WHEN OTHERS THEN

RAISE NOTICE 'An error occured: %', SQLERRM;

END

$$ LANGUAGE plpgsql;

--TO EXECUTE FUNCTION

SELECT fn\_patientsRecoveredByDrug('glatiramer');

-- Solution 2: With cursor

CREATE OR REPLACE FUNCTION fn\_patient\_recovered(drug text, patient\_cursor refcursor)

RETURNS refcursor

AS $$

BEGIN

RAISE NOTICE USING MESSAGE =

'LOOKING FOR THE PATIENTS RECOVERED USING THIS DRUG '||

drug;

OPEN patient\_cursor FOR

SELECT COUNT(c.patient\_id) as no\_Of\_Patient\_Recovered

FROM covid\_details c

RIGHT JOIN patient\_msdetails m using(patient\_id)

WHERE covid19\_outcome\_recovered=1

AND prescribed\_drug = drug

GROUP BY prescribed\_drug;

RETURN patient\_cursor;

END;

$$ LANGUAGE plpgsql;

---TO EXECUTE FUNCTION PLEASE RUN EACH STEP SEPERATELY.

BEGIN

SELECT fn\_patient\_recovered('latiramer','pcursor');

FETCH ALL FROM pcursor;

END;

BEGIN

SELECT fn\_patient\_recovered('dimethyl fumarate','pcursor');

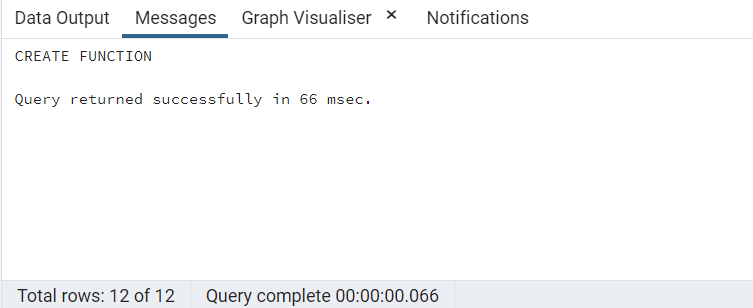
FETCH ALL IN pcursor;

END;

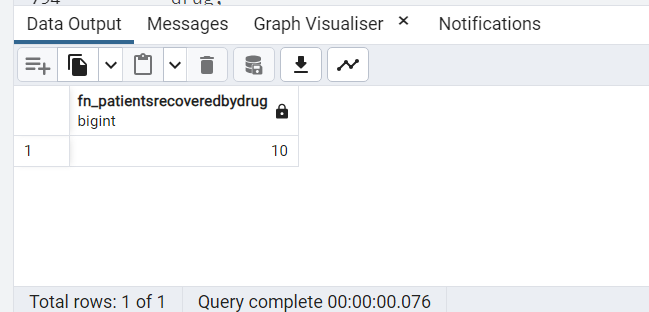
**Result:**

-- Solution 1: Without cursor

–create function

****

--TO EXECUTE FUNCTION

****

**Question 41:**

How many patients have drug names with length >7 letters?

**Query:**

SELECT

SUM(COUNT(patient\_id)) OVER() AS Patients\_drugname\_length\_gt7

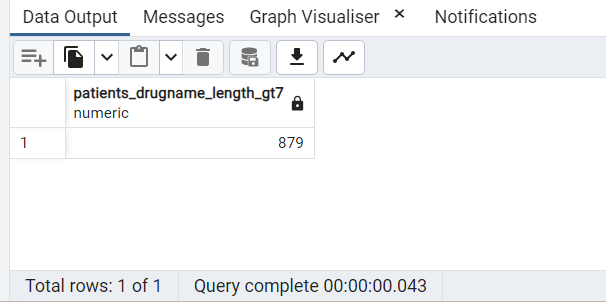
FROM

patient\_msdetails

WHERE

LENGTH(prescribed\_drug)>7;

**Result:**

****

**Question 42:**

Show the moving average of patients every 3 months.

**Query:**

SELECT

move\_avg.monYr AS YYYY\_MM,move\_avg.patients AS patient\_count,

round(AVG(move\_avg.patients) OVER

(ORDER BY move\_avg.monYr ROWS BETWEEN 2 PRECEDING AND CURRENT ROW),2)

AS moving\_avg\_Of\_patients\_every3months

FROM

(

SELECT

COUNT(patient\_id) AS patients,

to\_char(date\_of\_onset, 'YYYY-MM') AS monYr

FROM

covid\_details

GROUP BY

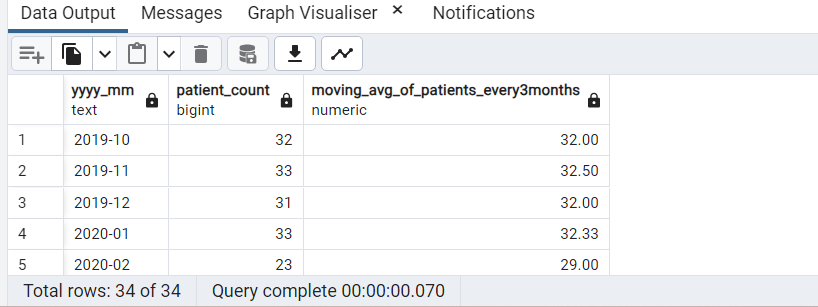
monYr

ORDER BY

monYr

) move\_avg;

**Result:**



**Question 43:**

What % of those who self isolated showed symptoms vs everyone else?

**Query:**

SELECT

MAX(CASE WHEN isolated.iso=1 THEN isolated.percentage

ELSE null END) AS percentageof\_isolated\_having\_symptoms,

MIN(CASE WHEN isolated.iso = 0 THEN isolated.percentage END) AS percentageof\_nonisolated\_having\_symptoms

FROM

(

SELECT

covid19\_self\_isolation AS iso,covid19\_has\_symptoms AS symp,

COUNT(patient\_id) as total,

to\_char((count(patient\_id) \* 100.0 / SUM(COUNT(patient\_id)) OVER (PARTITION BY covid19\_self\_isolation)),

'FM90.0" %"') AS Percentage

FROM covid\_details

GROUP BY covid19\_self\_isolation,covid19\_has\_symptoms

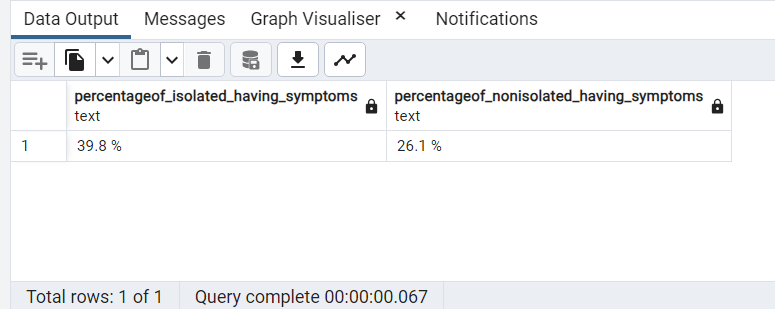
ORDER BY covid19\_has\_symptoms DESC

)isolated

WHERE

isolated.symp=1;

**Result:**

****

**Question 44:**

Create a trigger to stop patient records from being deleted from the patient details table

**Query:**

--Function

CREATE OR REPLACE FUNCTION stop\_delete\_patient\_detail\_tbl()

RETURNS TRIGGER AS $$

DECLARE

BEGIN

RAISE NOTICE 'TG\_OP IS %', TG\_OP;

IF TG\_OP='DELETE' THEN

IF EXISTS (SELECT \* FROM patient\_details WHERE patient\_id = OLD.patient\_id) THEN

RAISE EXCEPTION 'Record cannot be deleted';

END IF;

RETURN OLD;

else

RAISE EXCEPTION 'Trigger encountered unknown TG\_OP ';

RETURN OLD;

END IF;

END;

$$ LANGUAGE plpgsql;

---created trigger

CREATE OR REPLACE TRIGGER stop\_delete\_trigger

BEFORE DELETE ON patient\_details

FOR EACH ROW

EXECUTE FUNCTION stop\_delete\_patient\_detail\_tbl();

---try to delete record from 'covid\_detail\_temp\_tbl'

DELETE FROM patient\_details

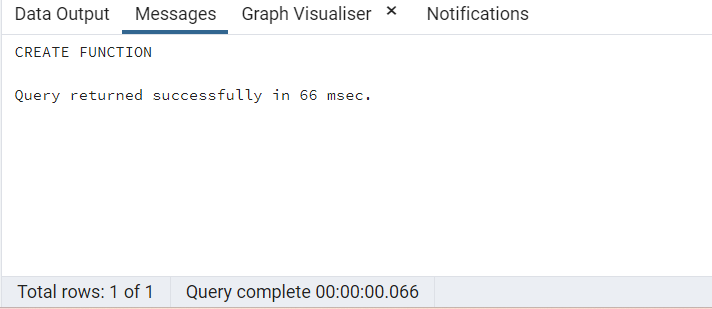
WHERE PATIENT\_ID = 'C\_1005';

-- Rollback the transaction for running next query

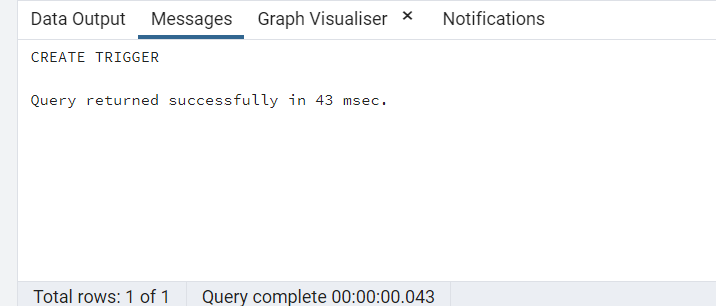
COMMIT;

**Result:**

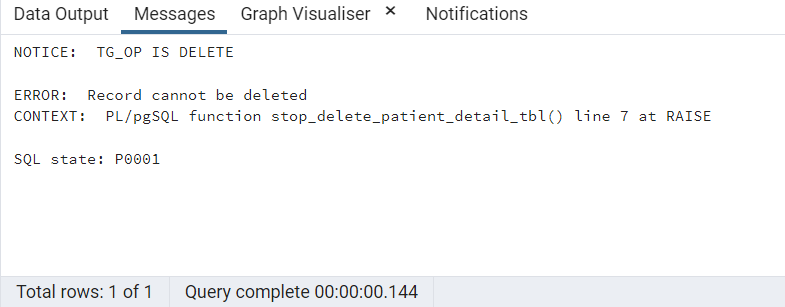
--Function

****

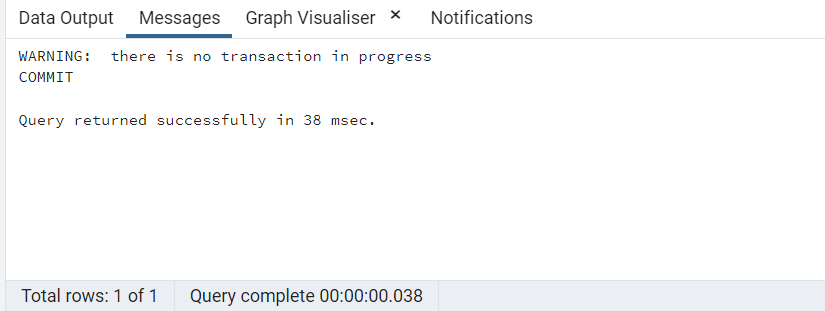
---created trigger

****

---try to delete record from 'covid\_detail\_temp\_tbl'

****

-- Rollback the transaction for running next query

****

**Question 45:**

Which age-group has the highest count of patients?

**Query:**

-- run commit to rollback the delete trigger transaction before running the next query

COMMIT;

SELECT high\_agegroup.age\_group

FROM

(SELECT

COUNT(patient\_id) AS cnt,

age\_group,Row\_number() OVER (ORDER BY count(patient\_id)DESC) AS rowNo

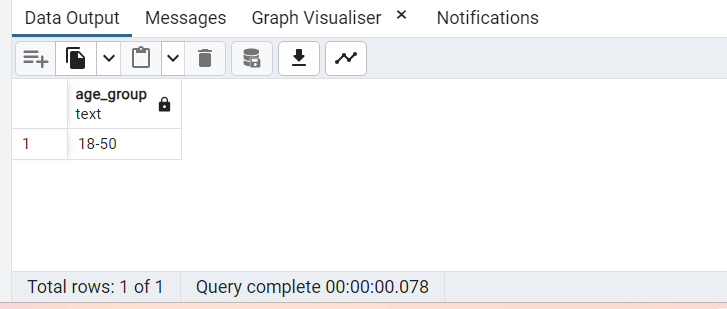
FROM patient\_details

GROUP BY age\_group) high\_agegroup

WHERE

high\_agegroup.rowNo=1;

**Result:**

****

**Question 46:**

Extract only the word fumarate from dimethyl fumarate and show it against the average number of years of treatment

**Query:**

SELECT

substr(prescribed\_drug,POSITION('fumarate' IN prescribed\_drug),LENGTH(prescribed\_drug)) AS drug,

AVG(duration\_treatment\_cat2)::numeric(10,5) AS avg\_years\_of\_treatment

FROM

patient\_msdetails

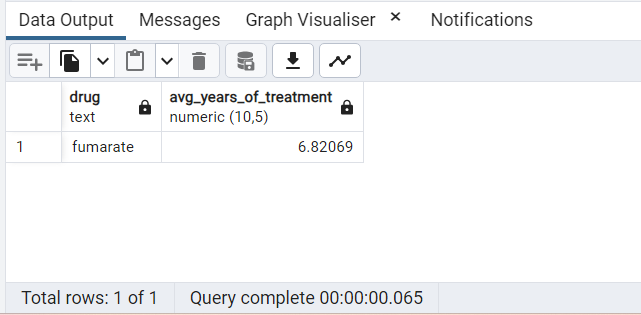
WHERE

prescribed\_drug = 'dimethyl fumarate'

GROUP BY

prescribed\_drug;

**Result:**

****

**Question 47:**

Write a query to display all drugs prescribed with Case-Insensitive Replacement of 'Unlisted' to 'Not Found'

**Query:**

SELECT DISTINCT(

(CASE WHEN prescribed\_drug ilike '%unlisted%'

THEN 'Not Found' ELSE prescribed\_drug END)

) AS prescribed\_drug

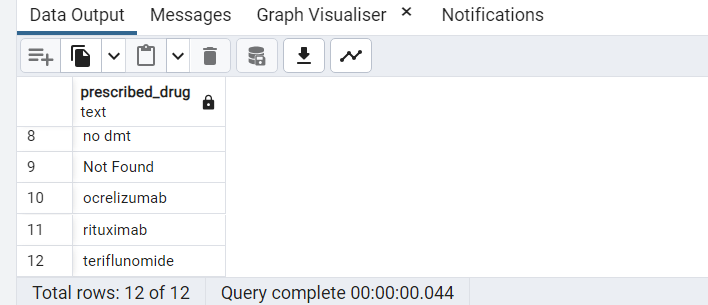
FROM

patient\_msdetails

ORDER BY

prescribed\_drug;

**Result:**

****

**Question 48:**

Create a table with patientID, BMI and Covid Status using a stored procedure. Add an auto generated sequence as the first column

**Query**:

-- stored procedure to create a table

CREATE OR REPLACE PROCEDURE proc\_covid()

LANGUAGE plpgsql

AS $$

BEGIN

CREATE TABLE status\_covid(

S\_id SERIAL PRIMARY KEY,

PatientID INT,

BMI INT ,

CovidStatus TEXT

);

RAISE NOTICE 'Table Created and Auto Generated Series is added to S\_Id';

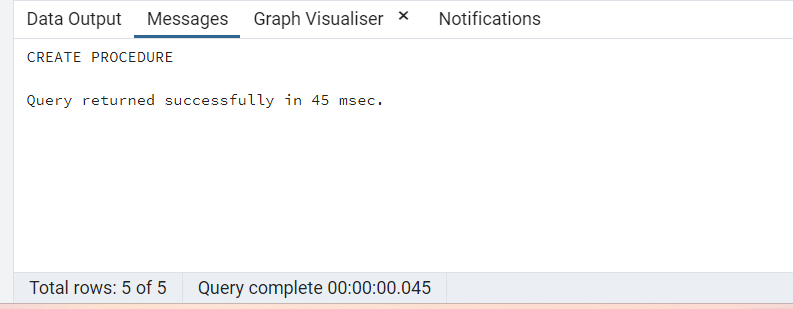
END;

$$;

-- call the stored procedure

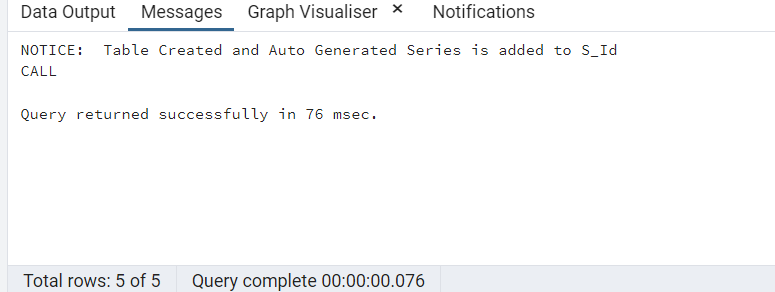
CALL proc\_covid();

**Result**:



-- call the stored procedure

CALL proc\_covid();



**Question 49:**

List patients where the MS Type ‘ssi’ along with the position where ‘ssi’ appears.

**Query**:

SELECT

patient\_id, POSITION('ssi' IN ms\_type2) AS position\_ssi

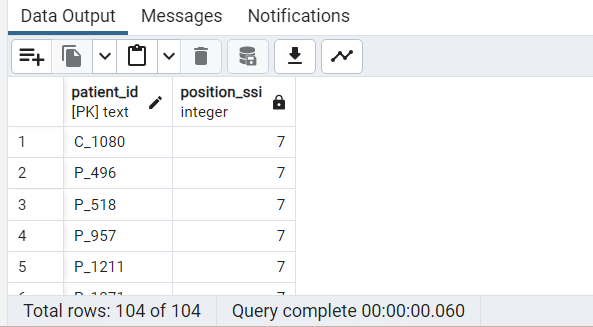
FROM

patient\_details

WHERE

ms\_type2 LIKE '%ssi%';

**Result**:



**Question 50:**

Create a child table for the table in Q48 which inherits all the values of the patient table and contains covid symptoms.

--ensure that at least 1 row is inserted into the final table

**Query**:

-- solution to create a child table and insert records

DO $$

BEGIN

IF EXISTS (SELECT 1 FROM information\_schema.tables WHERE table\_schema = 'public' AND table\_name = 'covstatchild') THEN

RAISE NOTICE 'Table already exists , so deleting and creating the new table.';

DROP TABLE covstatchild;

CREATE TABLE covstatchild (

CovidSymptoms INT

) INHERITS (status\_covid);

INSERT INTO covstatchild (PatientID, BMI, CovidStatus, CovidSymptoms)

VALUES (6, 12, 'Positive', 1), (8, 19, 'Negative', 0);

ELSE

RAISE NOTICE 'No table found, creating new table';

CREATE TABLE covstatchild (

CovidSymptoms INT

) INHERITS (status\_covid);

INSERT INTO covstatchild (PatientID, BMI, CovidStatus, CovidSymptoms)

VALUES (10, 11, 'Positive', 1), (9, 21, 'Negative', 0);

END IF;

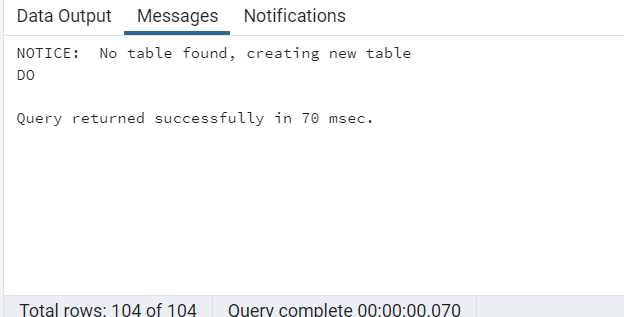
END;

$$;

-- select to view results

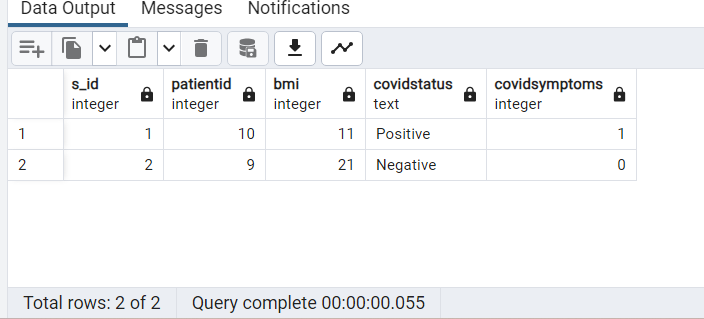
SELECT \* FROM covstatchild;

**Result**:



-- select to view results

SELECT \* FROM covstatchild;



**Question 51:**

List out any 5 patients, add '0' to their first part of the ID so that total characters displayed is 10

**Query:**

WITH PatientsList AS(

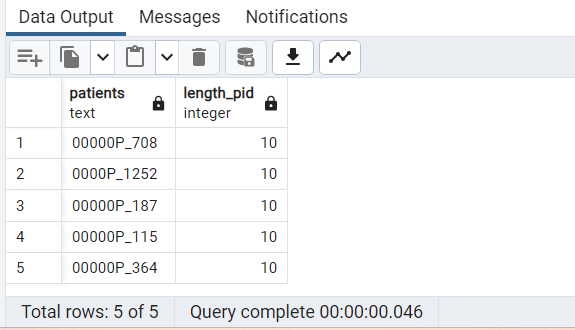
SELECT LPAD(patient\_id,10,'0') as Patients FROM patient\_details

)

SELECT Patients, Length(Patients) as Length\_PID FROM PatientsList

ORDER BY RANDOM() LIMIT 5

**Result:**

****

**Question 52:**

Write a trigger after inserting it on the Patient Details table. if the BMI >40, warn for high risk of covid symptoms

**Query:**

-- create a function to return a trigger

CREATE OR REPLACE FUNCTION Pat\_Det\_trigger\_fnc()

RETURNS TRIGGER AS

$$

BEGIN

IF NEW.bmi\_in\_cat2 > 40 THEN

RAISE NOTICE 'Patient is in high risk of covid symptoms, Inserting data with %bmi\_in\_cat2', NEW.bmi\_in\_cat2;

END IF;

RETURN NEW;

END;

$$

LANGUAGE 'plpgsql';

-- create a trigger when a row is inserted

CREATE TRIGGER Pat\_Det\_Aft\_Trigger

AFTER INSERT

ON patient\_details

FOR EACH ROW

EXECUTE PROCEDURE Pat\_Det\_trigger\_fnc();

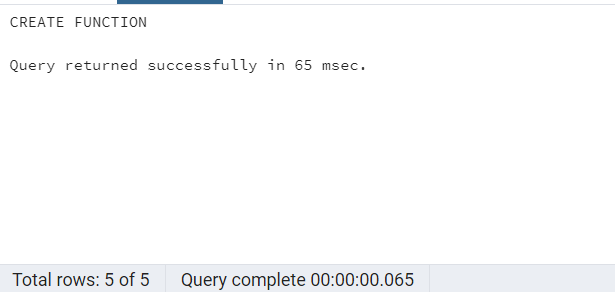
-- insert data to verify the results.

INSERT INTO patient\_details Values('P\_2424', 'patient', 0, 48, 'female', 'other', 0, 0, '18-50');

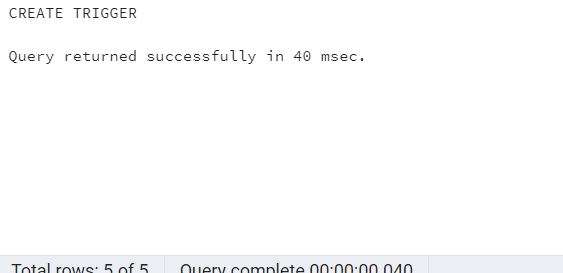
INSERT INTO patient\_details Values('P\_1919', 'patient', 1, 36, 'male', 'other', 0, 0, '18-50');

**Result:**

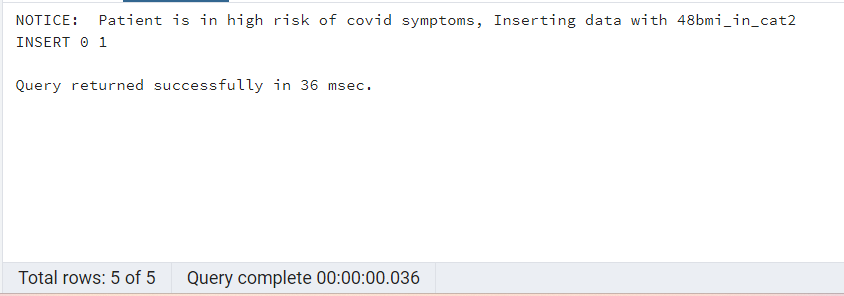
-- create a function to return a trigger

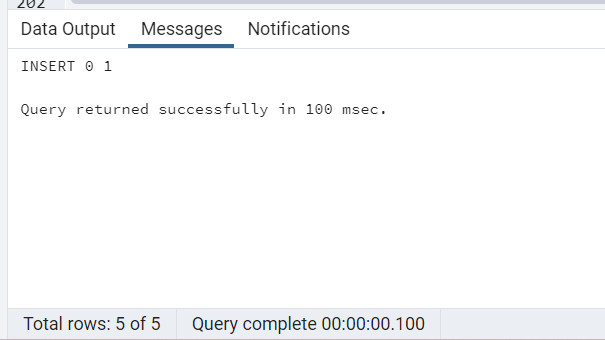
****

-- create a trigger when a row is inserted

****

-- insert data to verify the results.

****

****

**Question 53:**

Show the MS type of any patient in reverse

**Query:**

-- function to reverse ms\_type. Input is patient ID

CREATE OR REPLACE FUNCTION rev\_patient (patid text)

RETURNS TABLE (mstype2 text)

LANGUAGE plpgsql

AS $$

BEGIN

RETURN QUERY(

SELECT REVERSE(ms\_type2) FROM patient\_details

WHERE patient\_id = patid

);

END;

$$;

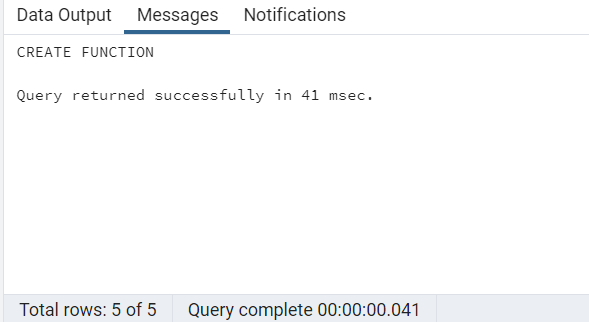
-- select statement to verify the results

SELECT \* FROM rev\_patient('P\_1010');

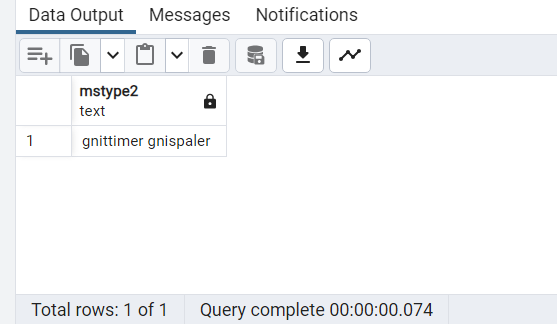
SELECT \* FROM rev\_patient('P\_123');

**Result:**

-- function to reverse ms\_type. Input is patient ID

****

-- select statement to verify the results

****

**Question 54:**

Create a pie chart to show the distribution of at least 5 symptoms across patients.

**Query:**

SELECT COUNT(patient\_id), Five\_Symptoms FROM

(

SELECT patient\_id,

CASE

WHEN covid19\_sympt\_pneumonia = 1 THEN 'Pnemonia'

WHEN covid19\_sympt\_dry\_cough = 1 THEN 'Dry\_Cough'

WHEN covid19\_sympt\_sore\_throat = 1 THEN 'Sore Throat'

WHEN covid19\_sympt\_loss\_smell\_taste = 1 THEN 'Loss\_Smell\_Taste'

WHEN covid19\_sympt\_shortness\_breath = 1 THEN 'Shortness of Breath'

ELSE 'Null'

END AS Five\_Symptoms FROM covid\_symptoms

)

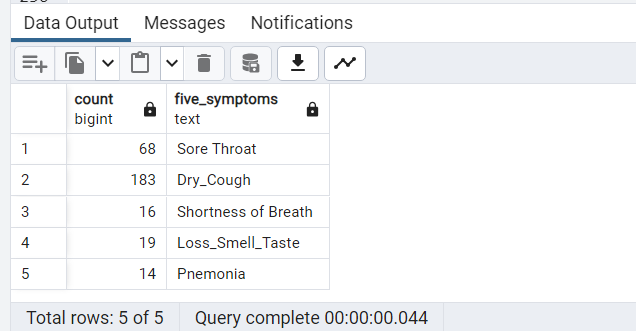
GROUP BY

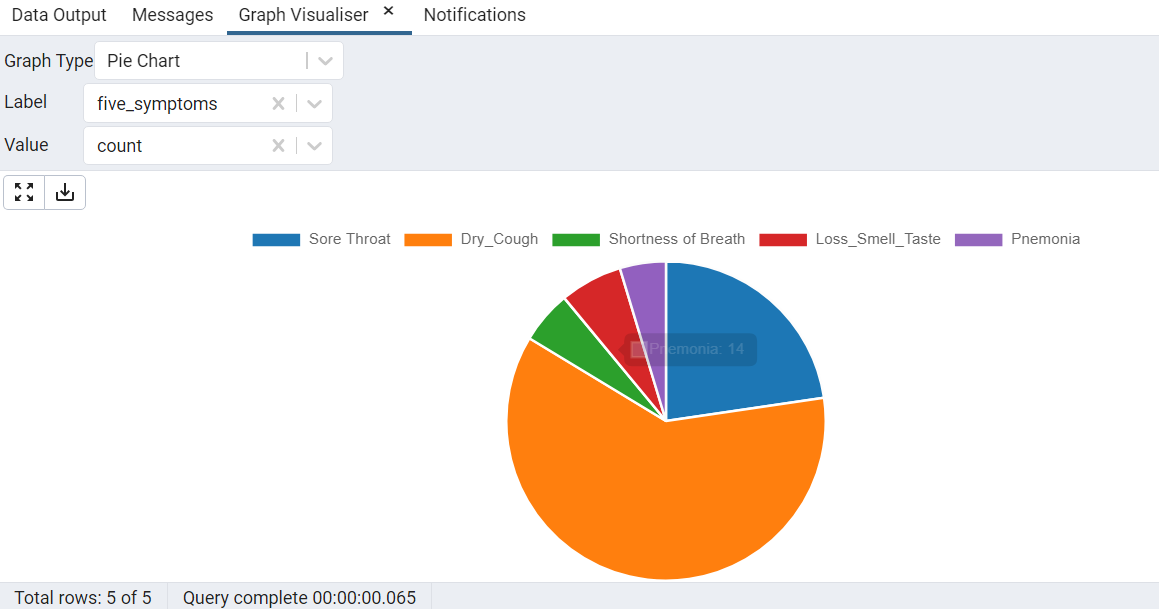
Five\_Symptoms

HAVING

Five\_Symptoms <> 'Null';

**Result:**

****



**Question 55:**

How many patients complained of pain as one of the symptoms of COVID.

**Query:**

SELECT

COUNT(patient\_id) AS Total\_Patients

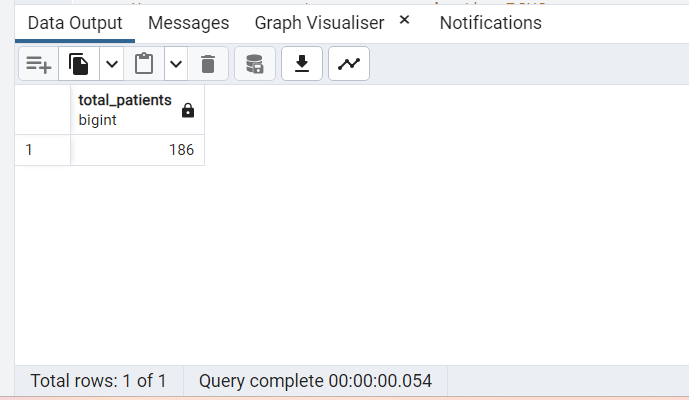
FROM

covid\_symptoms

WHERE

covid19\_sympt\_pain = 1;

**Result:**



**Question 56:**

Create a view that stores MS Type and any 4 comorbidities

**Query:**

DO $$

BEGIN

IF NOT EXISTS (SELECT 1 FROM information\_schema.views WHERE table\_name = 'MStype\_Com') THEN

-- Create the view here

CREATE OR REPLACE VIEW MStype\_Com AS

SELECT ms\_type2, com\_lung\_disease, com\_hypertension, com\_malignancy,com\_diabetes FROM patient\_details PD

INNER JOIN comorbidities CO

ON PD.patient\_id = CO.patient\_id;

RAISE NOTICE 'View created';

ELSE

RAISE NOTICE 'View already exists. Deleting';

DROP VIEW MStype\_Com;

CREATE OR REPLACE VIEW MStype\_Comorbidities AS

SELECT ms\_type2, com\_lung\_disease, com\_hypertension, com\_malignancy,com\_diabetes from patient\_details PD

INNER JOIN comorbidities CO

ON PD.patient\_id = CO.patient\_id;

RAISE NOTICE 'View created';

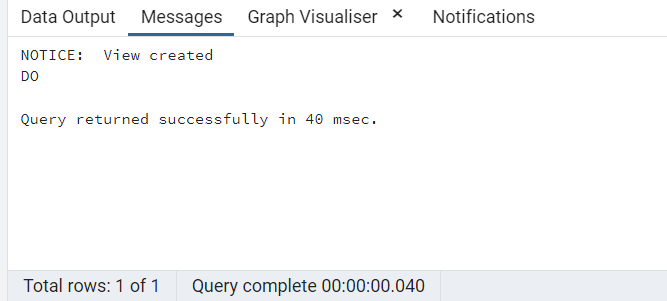
END IF;

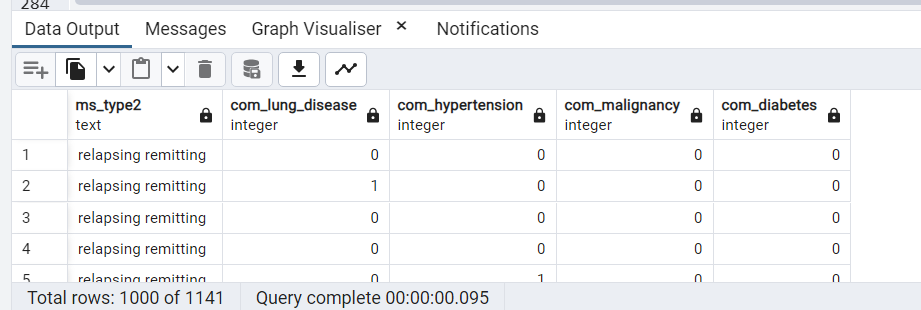
END $$;

-- Select to see results

SELECT \* FROM MStype\_Com

**Result:**

****

****

**Question 57:**

How many pregnant women are in the ICU?

**Query:**

SELECT

COUNT(PD.patient\_id) as No\_of\_Patients

FROM

patient\_details PD

JOIN

covid\_details CD

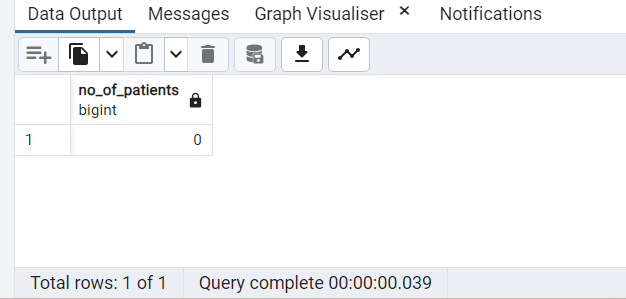
ON

CD.patient\_id = PD.patient\_id

WHERE

pregnancy = 1 AND covid19\_icu\_stay = 1;

**Result:**

****

**Question 58:**

Display the Day and Year Of onset in 2 separate columns for the patient who were on no Drugs

**Query:**

SELECT

PM.patient\_id, pm.prescribed\_drug,

EXTRACT(DAY FROM date\_of\_onset) AS Day,

EXTRACT(Year FROM date\_of\_onset) AS Year

FROM

covid\_details CD

INNER JOIN

patient\_msdetails PM

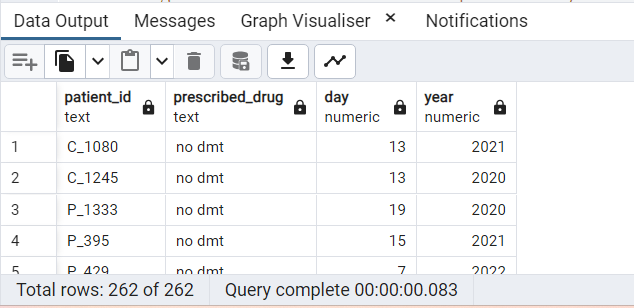
ON

PM.patient\_id=CD.patient\_id

WHERE

prescribed\_drug = 'no dmt';

**Result:**

****

**Question 59:**

What % of the dataset has been confirmed with a diagnosis of covid

**Query:**

SELECT

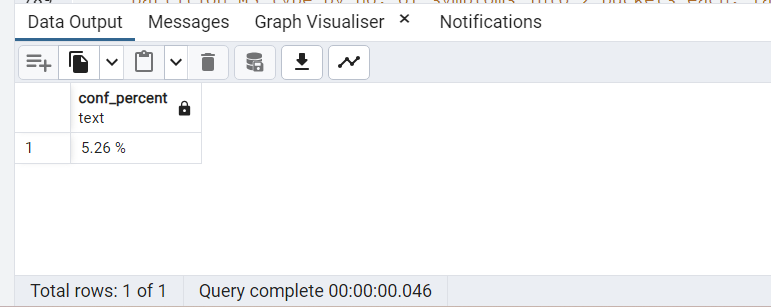
to\_char((Conf\_Count \* 100.0 / Total\_count), 'FM90.00" %"') AS Conf\_percent

FROM

(SELECT COUNT(covid19\_diagnosis) Conf\_Count FROM covid\_details WHERE covid19\_diagnosis ='confirmed'),

(SELECT COUNT(\*) Total\_count FROM covid\_details);

**Result:**

****

**Question 60:**

Using Windows functions. Divide patients by MS type,

--partition MS\_type by no. of symptoms into 2 buckets each, label them as high and low risk. Display avg symptoms in each bucket

**Query:**

WITH MStype\_symptoms AS

(SELECT patient\_id,

SUM(covid19\_sympt\_fatigue + covid19\_sympt\_fever + covid19\_sympt\_pain + covid19\_sympt\_sore\_throat +

covid19\_sympt\_chills + covid19\_sympt\_dry\_cough + covid19\_sympt\_loss\_smell\_taste +

covid19\_sympt\_nasal\_congestion + covid19\_sympt\_pneumonia + covid19\_sympt\_shortness\_breath

) AS Total\_symp FROM covid\_symptoms

GROUP BY patient\_id

),

GroupBucket AS (

SELECT patient\_id,

CASE

WHEN Total\_symp BETWEEN 0 AND 5 THEN 'Low Risk'

WHEN Total\_symp BETWEEN 6 AND 10 THEN 'High Risk'

END AS Bucket\_Category FROM MStype\_symptoms MS

GROUP BY patient\_id, Total\_symp

)

SELECT

ms\_type2, COUNT(MS.patient\_id) AS Patient\_count, Total\_symp, Bucket\_Category,

ROUND(AVG(Total\_symp) OVER(partition by Bucket\_Category),2) AS Avg\_symp

FROM MStype\_symptoms MS

JOIN GroupBucket B

ON MS.patient\_id = B.patient\_id

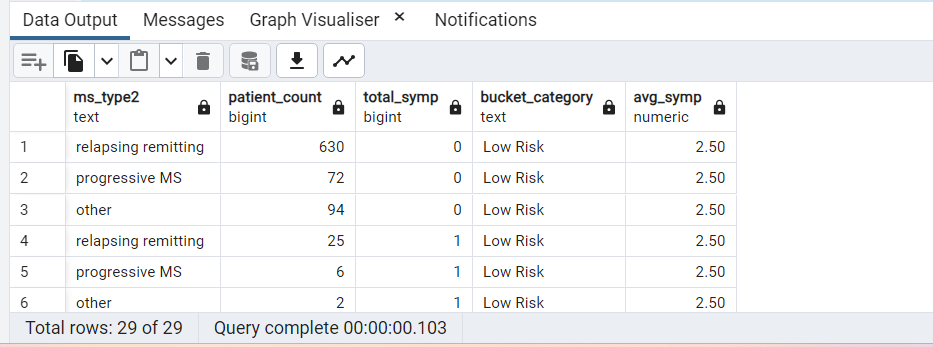
JOIN patient\_details PD

ON MS.patient\_id = PD.patient\_id

GROUP BY ms\_type2, Total\_symp, Bucket\_Category

ORDER BY Total\_symp;

**Result:**

****

**Question 61:**

Using the ANY function, what is the maximum treatment duration observed among all patients?

**Query:**

SELECT

MAX(duration\_treatment\_cat2) AS Treatment\_duration

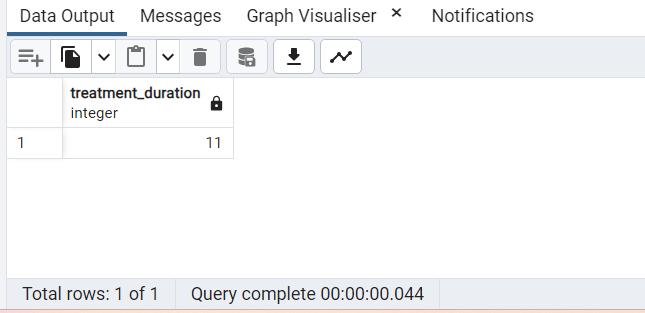
FROM

patient\_msdetails

WHERE

patient\_id = ANY(SELECT patient\_id FROM patient\_details);

**Result:**

****

**Question 62:**

Use Case statements to add meaning to the EDSS Score and show number of patients and covid diagnosis per group

**Query:**

SELECT

CASE

WHEN edss\_in\_cat2 ='0' THEN 'Normal Neuro Examination'

WHEN edss\_in\_cat2 >'0' AND edss\_in\_cat2 <='5' THEN 'Fully ambulatory patients'

WHEN edss\_in\_cat2 ='10' THEN 'MS-related death cases'

ELSE ' With Assistance'

END AS EDSS\_Score,

COUNT(cd.patient\_id) AS Patient\_Count,

COUNT(cd.covid19\_diagnosis) AS Covid\_diagnosis\_Count

FROM

patient\_msdetails p

JOIN

covid\_details cd

ON cd.patient\_id = p.patient\_id

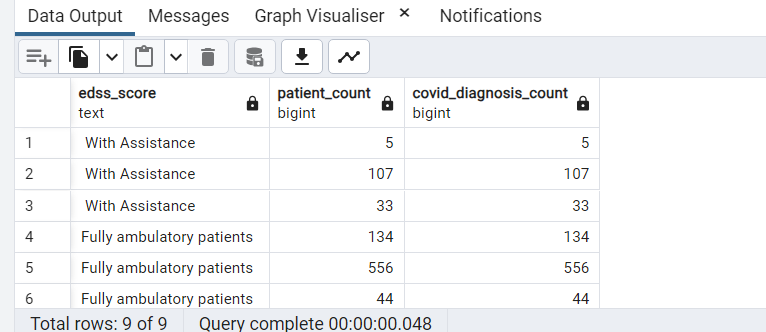
GROUP BY

cd.covid19\_diagnosis, EDSS\_Score

ORDER BY

EDSS\_Score;

**Result:**

****

**Question 63:**

Find the average treatment duration for patients and round up to the nearest integer value.

**Query:**

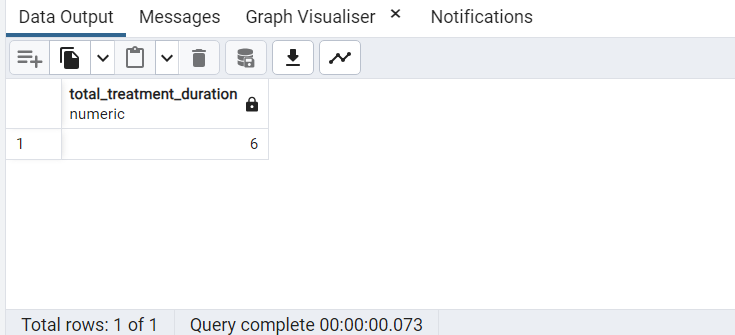
SELECT

ROUND(AVG(duration\_treatment\_cat2)) AS Total\_Treatment\_duration

FROM

patient\_msdetails;

**Result:**

****

**Question 64:**

Select the top 5 tables in the COVID database by size

**Query:**

SELECT

table\_name as CovidDB\_TableNames,pg\_relation\_size(table\_schema || '.' || table\_name) as CovidDB\_Size

FROM

information\_schema.tables

WHERE

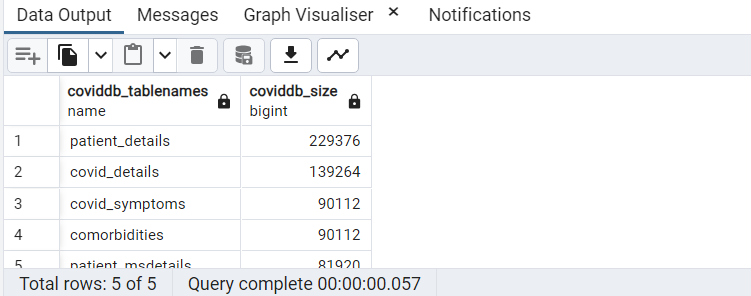
table\_schema NOT IN ('information\_schema', 'pg\_catalog')

ORDER BY

CovidDB\_Size DESC

LIMIT 5;

**Result:**

****

**Question 65:**

Divide EDSS by treatment duration for any 5 patients without using mathematical operators like '/'

**Query:**

-- The DIV() function accepts any positive values, negative values,

-- fractional/floating point values, etc. as arguments and retrieves an integer value

SELECT

patient\_ID,edss\_in\_cat2 AS EDSS,

duration\_treatment\_cat2 AS treatment\_duration,

DIV(edss\_in\_cat2,duration\_treatment\_cat2) AS div\_EDSS\_TreatmentDuration

FROM

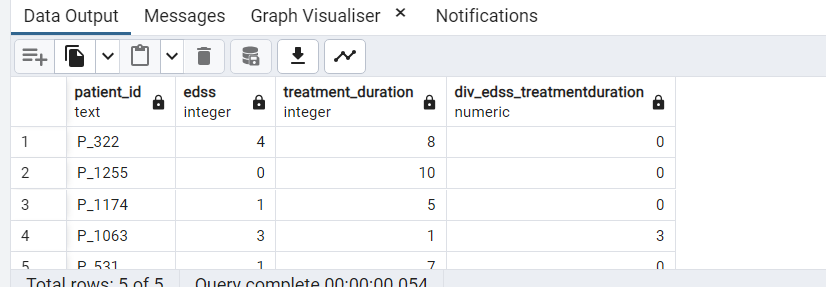
patient\_msdetails

ORDER BY

RANDOM()

LIMIT 5;

**Result:**

****

**Question 66:**

Create a trigger on a view from q56.

**Query:**

**Result:**

**Question 67:**

Divide EDSS by treatment duration for any 5 patients without using mathematical operators like '/' and return any remainder

**Query:**

-- MOD() function returns only the remainder after division

SELECT

patient\_ID,edss\_in\_cat2 AS EDSS,

duration\_treatment\_cat2 AS treatment\_duration,

MOD(edss\_in\_cat2,duration\_treatment\_cat2) AS remainder\_EDSS\_TreatmentDuration

FROM

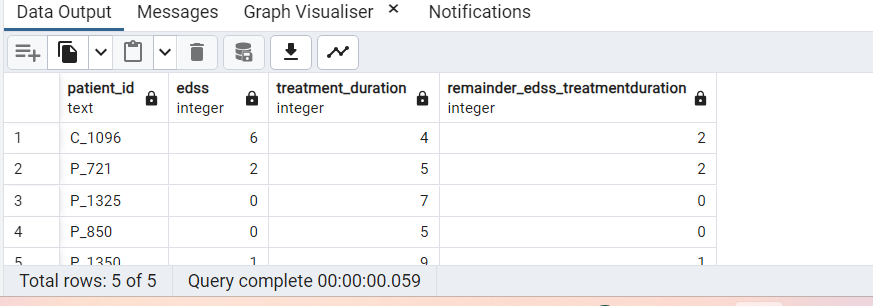
patient\_msdetails

ORDER BY

RANDOM()

LIMIT 5;

**Result:**

****

**Question 68:**

Provide the cumulative distribution for MS type based on average being BMI below or above 30.

**Query:**

**Result:**

**Question 69:**

What is the total number of years between the earliest record and the latest

**Query:**

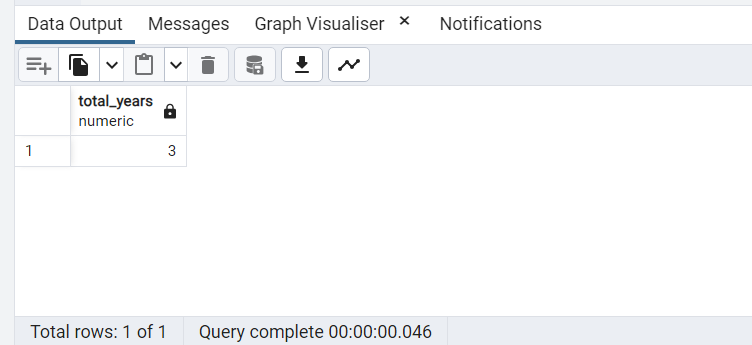
SELECT

EXTRACT(YEAR FROM MAX(date\_of\_onset)) - EXTRACT(YEAR FROM MIN(date\_of\_onset)) AS total\_years

FROM

covid\_details;

**Result:**

****

**Question 70:**

What % of those on glucocorticoids have symptoms and what is the most common symptom among this population?

**Query:**

--Functions: ARRAY, LATERAL

SELECT e.percsympt as percentage\_on\_glucocorticoids\_have\_symptoms,

d.row AS most\_common\_symptom FROM (

SELECT SUM(sympArr.elem), sympArr.rowNo

,CASE

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 1 THEN 'covid19\_sympt\_chills'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 2 THEN 'covid19\_sympt\_fatigue'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 3 THEN 'covid19\_sympt\_dry\_cough'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 4 THEN 'covid19\_sympt\_fever'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 5 THEN 'covid19\_sympt\_loss\_smell\_taste'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 6 THEN 'covid19\_sympt\_pain'

WHEN to\_char((MAX(SUM(sympArr.elem)) OVER()),'FM90')::integer = SUM(sympArr.elem)

AND sympArr.rowNo = 7 THEN 'covid19\_sympt\_pneumonia'

ELSE 'null' END AS ROW

FROM

-- select s.patient\_id,a.arr from (

COVID\_DETAILS c

JOIN PATIENT\_MSDETAILS USING (PATIENT\_ID)

JOIN COVID\_SYMPTOMS S USING (PATIENT\_ID) --)

,LATERAL (SELECT ARRAY[

covid19\_sympt\_chills,covid19\_sympt\_fatigue,

covid19\_sympt\_dry\_cough,covid19\_sympt\_fever,

covid19\_sympt\_loss\_smell\_taste,covid19\_sympt\_pain,

covid19\_sympt\_pneumonia] AS arr) a

, UNNEST(a.arr) WITH ORDINALITY sympArr(elem,rowNo)

WHERE DMT\_GLUCOCORTICOID =1

AND 1 = ANY(arr)

GROUP BY rowNo

ORDER BY rowNo

)d

, (SELECT COUNT(s.patient\_id),covid19\_has\_symptoms

,to\_char((COUNT(patient\_id)\*100.0/SUM(COUNT(patient\_id))

OVER ()), 'FM90.00"%"')AS percsympt

FROM

COVID\_DETAILS c

JOIN PATIENT\_MSDETAILS USING (PATIENT\_ID)

JOIN COVID\_SYMPTOMS S USING (PATIENT\_ID)

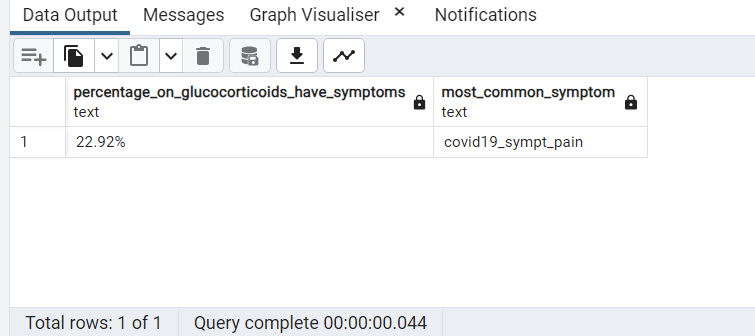
WHERE DMT\_GLUCOCORTICOID =1

GROUP BY covid19\_has\_symptoms) e

WHERE row != 'null'

AND e.covid19\_has\_symptoms=1

**Result:**

****

**Question 71:**

Display the patients with the highest BMI.

**Query:**

SELECT \*

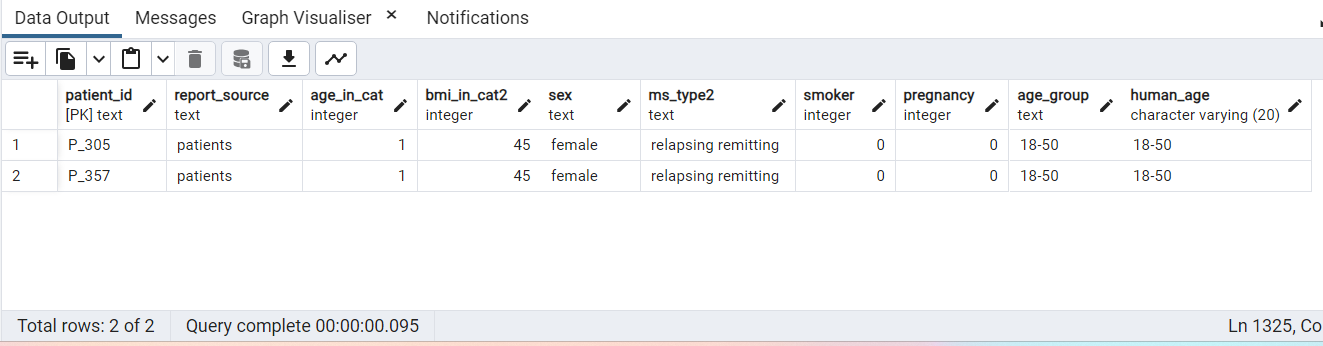
FROM

patient\_details

WHERE

bmi\_in\_cat2 = (SELECT MAX(bmi\_in\_cat2) FROM patient\_details);

**Result:**

****

**Question 72:**

Is there a correlation between EDSS and Recovery or Ventilation? Show what correlation exists.

**Query:**

-- correlation between edss and Ventilation

SELECT

round(CORR(pm.edss\_in\_cat2, cd.covid19\_ventilation)::numeric,2) AS correlation\_edss\_ventilation

FROM

patient\_msdetails pm

JOIN

covid\_details cd

ON

pm.patient\_id = cd.patient\_id;

-- correlation between edss and recovery

SELECT

round(CORR(pm.edss\_in\_cat2, cd.covid19\_outcome\_recovered)::numeric,2) AS correlation\_edss\_recovery

FROM

patient\_msdetails pm

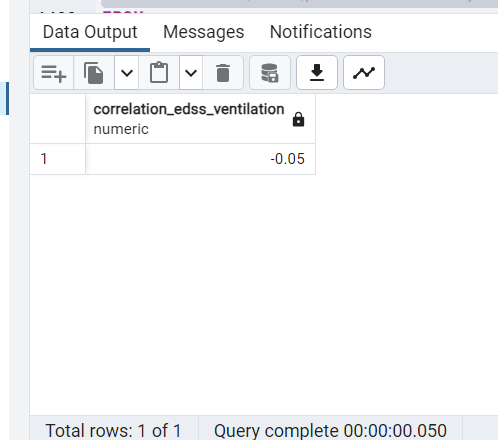
join

covid\_details cd

ON

pm.patient\_id = cd.patient\_id;

**Result:**

****

**Question 73:**

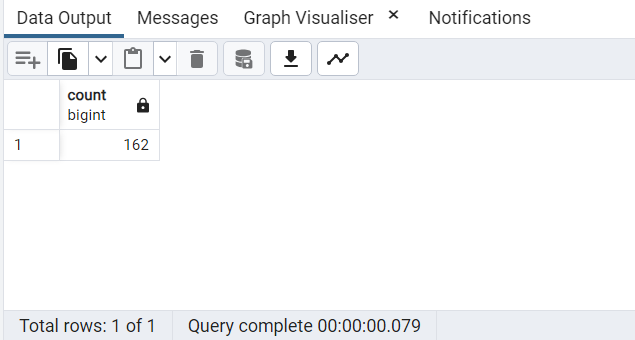
How many patients in the dataset have a BMI of 30 or more?

**Query:**

SELECT COUNT(patient\_id) FROM patient\_details

WHERE bmi\_in\_cat2 >=30;

**Result:**

****

**Question 74:**

How many patients in the ICU had more than 3 comorbidities?

**Query:**

WITH patient\_ICU AS

(

SELECT

cd.patient\_id,

(SELECT SUM(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease +

com\_diabetes + com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy +

com\_neurological\_neuromuscular) AS patient\_comorbidities

FROM comorbidities como

WHERE cd.patient\_id = como.patient\_id)

FROM

covid\_details cd

JOIN

comorbidities com

ON

cd.patient\_id = com.patient\_id

WHERE

covid19\_icu\_stay = 1

ORDER BY

patient\_comorbidities DESC

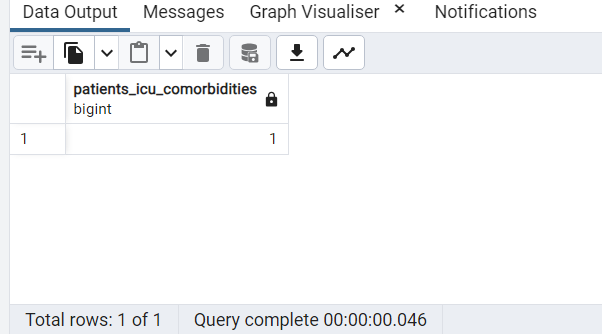
)

SELECT COUNT(patient\_id) as patients\_ICU\_comorbidities

FROM patient\_ICU

WHERE patient\_comorbidities > 3

**Result:**

****

**Question 75:**

Alter table patient\_msdetails change the column type of edss\_in\_cat2 to text and retain all values.

--Reverse the change and re-update it as an integer without losing any values

**Query:**

-- Step 1: Change the column type to text

ALTER TABLE patient\_msdetails

ALTER COLUMN edss\_in\_cat2 TYPE TEXT;

–verifying using select

SELECT \* FROM patient\_msdetails

-- Step 2: Reverse the column type to integer

ALTER TABLE patient\_msdetails

ALTER COLUMN edss\_in\_cat2 TYPE INTEGER

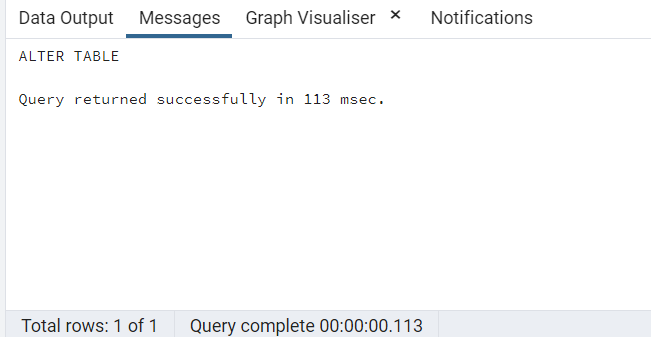
USING (edss\_in\_cat2::INTEGER);

–verifying using select

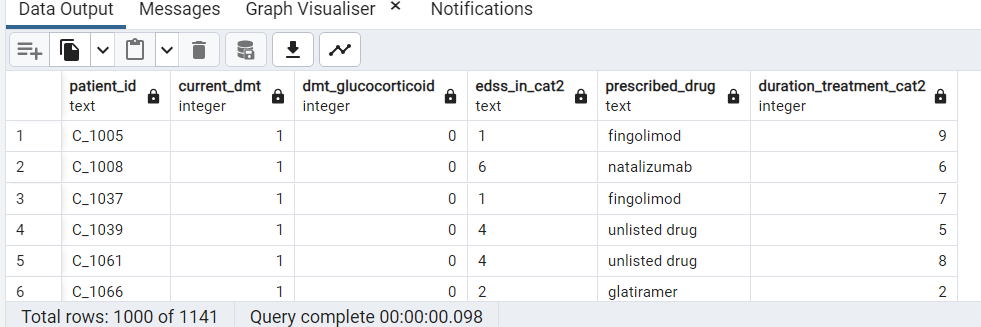
SELECT \* FROM patient\_msdetails

**Result:**

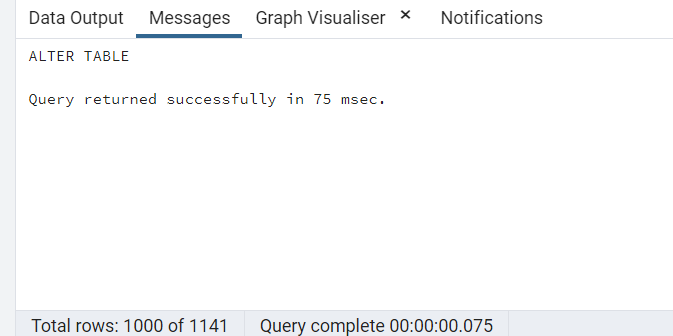
-- Step 1: Change the column type to text

****

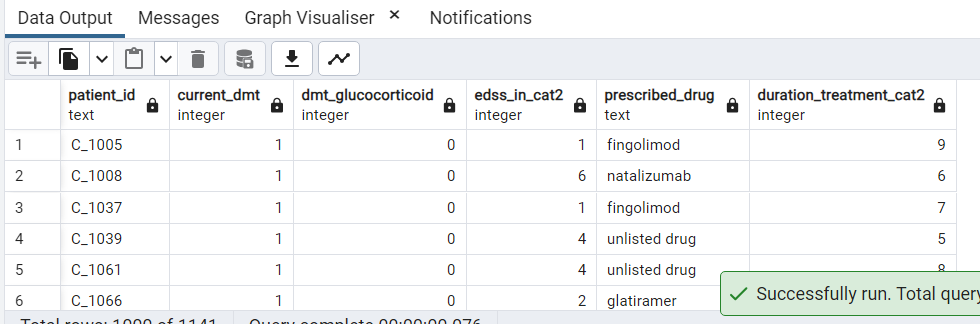
–verifying using select

****

-- Step 2: Reverse the column type to integer

****

–verifying using select

****

**Question 76:**

Write a query using recursive view(use the given dataset only)

**Query:**

--created recursive view to find patients those are admitted in the hospital

--in the same month (year may differ) for date\_of\_onset for patient 'C\_1005'

CREATE OR REPLACE RECURSIVE VIEW recursive\_view (patient,patient1,dt) AS

SELECT patient\_id AS patient, 'patient1' AS patient1

,DATE\_PART('month',date\_of\_onset) AS dt

FROM

covid\_details

where patient\_id = 'C\_1005'

UNION ALL

SELECT

c.patient\_id AS patient1, recursive\_view.patient AS patient

,DATE\_PART('month',date\_of\_onset) AS dt

FROM

covid\_details c

INNER JOIN recursive\_view ON DATE\_PART('month',date\_of\_onset) = recursive\_view.dt

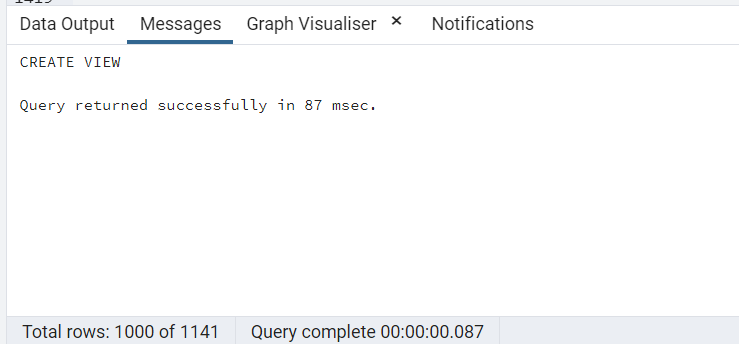
WHERE covid19\_admission\_hospital=1

–verification using select

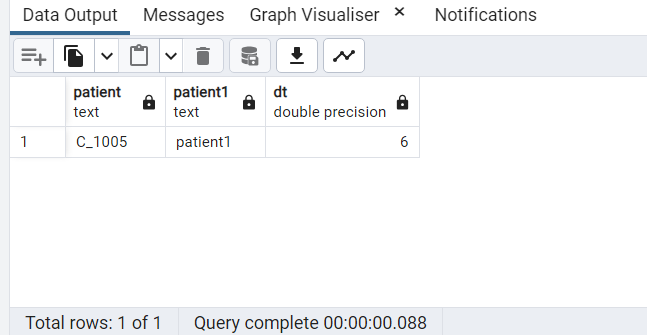
SELECT \* FROM recursive\_view

**Result:**

--created recursive view to find patients those are admitted in the hospital

****

–verification using select

****

**Question 77:**

What is the 3rd highest BMI of the patients with more than 2 comorbidities? Use windows functions?

**Query:**

WITH patient\_comorbidities AS

(

SELECT

pd.patient\_id,pd.bmi\_in\_cat2,

(SELECT sum(com\_cardiovascular\_disease + com\_chronic\_kidney\_disease + com\_chronic\_liver\_disease

+ com\_diabetes + com\_hypertension + com\_immunodeficiency + com\_lung\_disease + com\_malignancy

+ com\_neurological\_neuromuscular) AS comorbidities\_SUM

FROM comorbidities como WHERE pd.patient\_id = como.patient\_id)

FROM

patient\_details pd

JOIN

comorbidities como

ON

pd.patient\_id = como.patient\_id

)

SELECT bmi\_in\_cat2 AS hightest\_3rdBMI\_having\_morethan\_2\_comorbidties

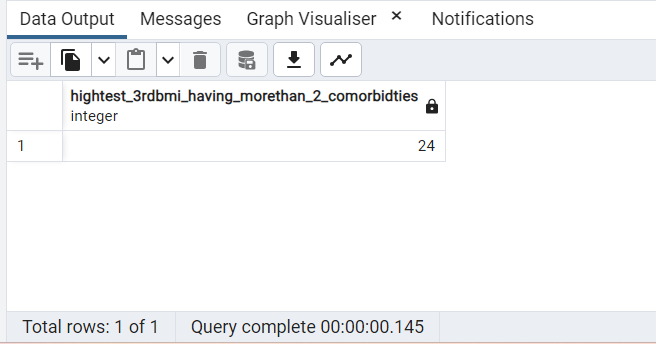
FROM (SELECT bmi\_in\_cat2,

DENSE\_RANK() OVER (ORDER BY bmi\_in\_cat2 DESC) AS denserank

FROM patient\_comorbidities WHERE comorbidities\_SUM > 2)

WHERE denserank = 3;

**Result:**



**Question 78:**

Update the table patient\_msdetails. Set drugs prescribed to be Sentence case, query the results of the updated table without writing a second query

**Query:**

UPDATE

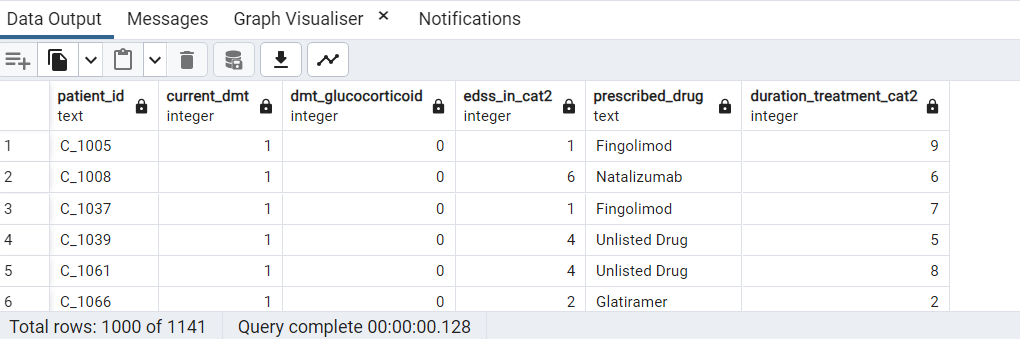
patient\_msdetails

SET

prescribed\_drug = INITCAP(prescribed\_drug)

RETURNING \*;

**Result:**

****

**Question 79:**

Use the materialized view in Q14. to show every patients details in one single line using a JSON function

**Query:**

-- Query to convert all rows into one single line

SELECT

array\_to\_json(ARRAY\_AGG(row\_to\_json(tablerow)))

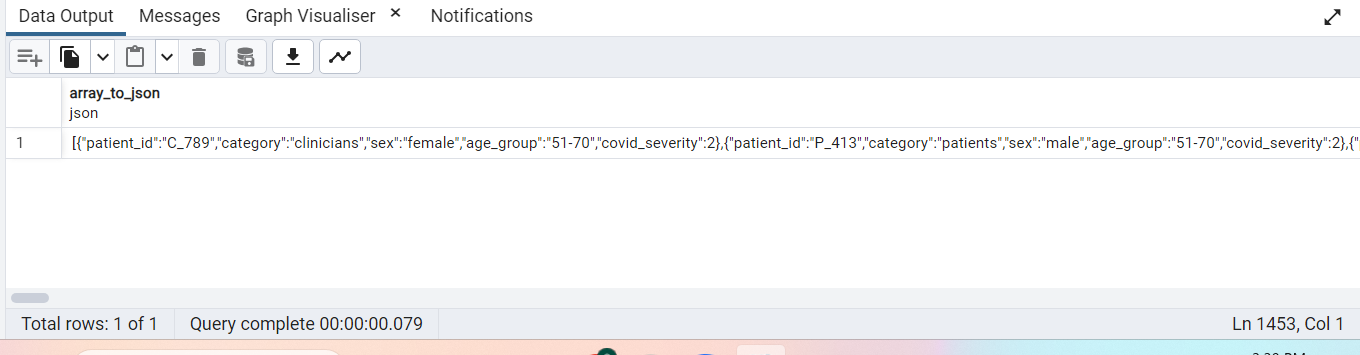
FROM (

SELECT patient\_id,category,sex,age\_group,covid\_severity

FROM view\_covidSeverity

) tablerow

**Result:**

****

**Question 80:**

Create a stored procedure that adds a column to table Covid Details.

--The column should just be the Year extracted from Date Onset

**Query:**

-- Drop the stored procedure "addcolumn\_updateyear" if exists

DROP PROCEDURE IF EXISTS addcolumn\_updateyear;

-- Drop the column "onset\_year" from the table covid\_details if exists

ALTER TABLE covid\_details

DROP COLUMN onset\_year;

-- Stored Procedure to add a new column and populate the values into that column

CREATE OR REPLACE PROCEDURE addcolumn\_updateyear(OUT Infomessage text)

LANGUAGE plpgsql

AS $$

BEGIN

-- Add a new column with the name "onset\_year" to covid\_details table

ALTER TABLE covid\_details

ADD COLUMN onset\_year INT;

-- Extract the year from "date\_of\_onset" column and updates the value to "onset\_year" column

UPDATE covid\_details

SET onset\_year = EXTRACT(YEAR FROM date\_of\_onset);

SELECT 'Updated Successfully' INTO Infomessage;

END; $$

-- Call to the stored procedure - returns a message after the execution is sucessful.

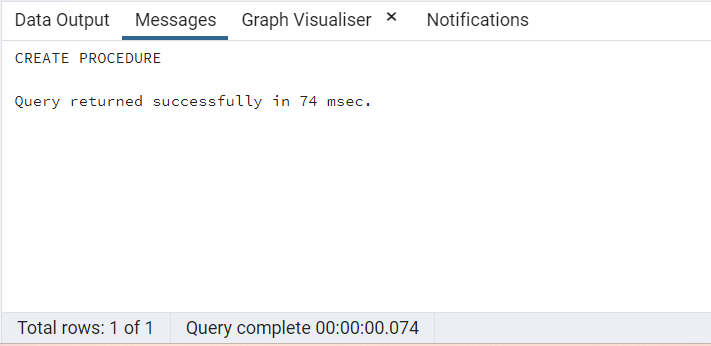
call addcolumn\_updateyear(NULL);

-- Select query to verify the results

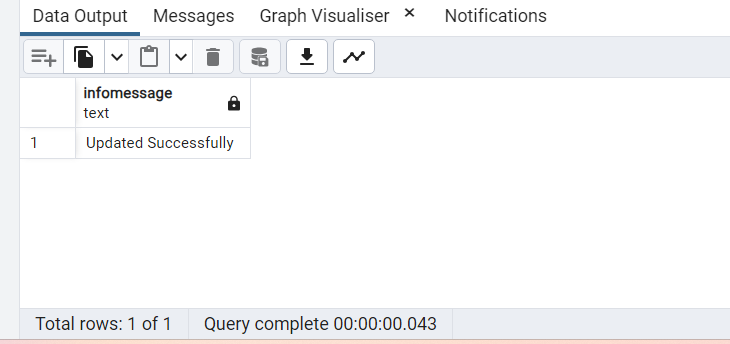
SELECT patient\_id,date\_of\_onset,onset\_year FROM covid\_details;

**Result:**

-- Stored Procedure to add a new column and populate the values into that column



-- Call to the stored procedure - returns a message after the execution is sucessful.



-- Select query to verify the results

