FIT5137 S2 2023 Assignment 4: PTV Answer Sheet (Weight = 30%)

PLEASE SUBMIT ANSWER SHEET IN PDF FORMAT

Due date: Wednesday, 25 October 2023, 11:55pm

Version: 2.0 – 25/09/2023

Assignment Task list

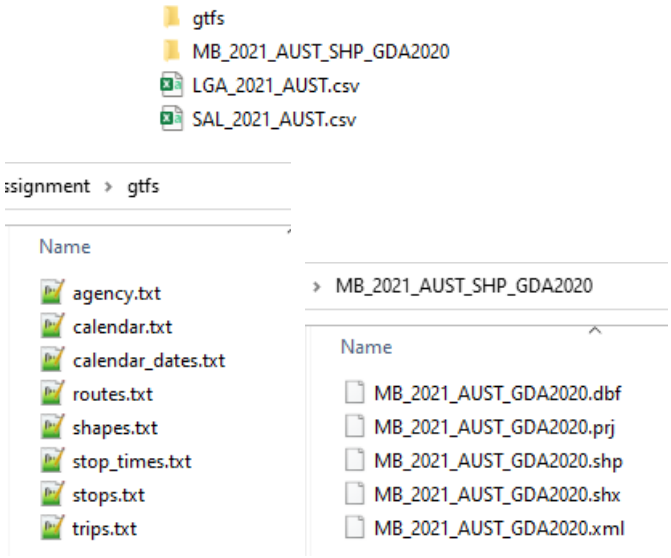
Your assignment consists of several parts. Always read the instruction one by one. Do not move to the step without completing the previous step:

- Task 1: Data Restoration - Restore the data to the database. Monitor the success indicator to ensure successful restoration of the data.
- Task 2: Data Preprocessing - Perform necessary structure maintenance and create result tables for further processing.
- Task 3: Data Analytics - Develop SQL queries to analyze the data and evaluate performance.
- Task 4: Data Visualization - Create visualizations to present the results of the data analytics.

For simplicity, all the data required for this assignment is readily available in the PostGIS Docker container. You can access these datasets within the container by navigating to the /data/adata folder.If you don't know how to do it, refer to the labs 10 activities. As a data analyst, it is your responsibility to understand and explore these publicly available data.

```
root@db94d38b7162:/home/student# ls /data/adata
gtfs  LGA_2021_AUST.csv  MB_2021_AUST_SHP_GDA2020  SAL_2021_AUST.csv
```

Verify your data before the restoration process.



As a data analyst, it is your responsibility to understand and explore these publicly available data.

Do not edit or remove any content on the answer sheet, including the questions. Please write your answers in the answer boxes provided. If necessary, you can adjust the size of the answer box to fit your answer.

Task 1: Data Restoration

Before you can start the data analytic processes, the first thing you have to do is to restore the external data to your database. Make sure you prepare a destination schema to restore your data. The destination schema for your assignment is "**ptv**".

Note:

- Before initiating the data restoration process, **it is essential to thoroughly explore the dataset**. This exploration involves identifying appropriate data types, determining field lengths, and making other relevant considerations that will inform the creation of the table structure.
- Ensure that you restore the data into the PTV schema using regular (local) tables. Do not utilize foreign tables, as the data must be stored directly within the PostgreSQL database – updated 27/09/2023
- Make sure all 8 GTFS tables are restored successfully
- Index or constraints can be added to the table after the data has been restored completely (Note: This index or constraint requirement is **NOT** mandatory in this Task 1 – updated 25/09/2023)
- **No data cleaning required for this assignment**
- For more information, see the FAQ for Assignment 4.

1.1 PTV schema

Write the SQL script to create the destination schema named "**ptv**".

```
create schema ptv;
```

1.2 GTFS

Write the SQL script to restore **ALL** tables in GTFS files.

```
-- Agency
-- drop table ptv.agency;

create table ptv.agency(
		agency_id numeric,
		agency_name varchar,
		agency_url varchar,
		agency_timezone varchar,
		agency_lang varchar
		);

copy ptv.agency(agency_id, agency_name, agency_url, agency_timezone, agency_lang)
from '/data/adata/gtfs/agency.txt'
delimiter ',' csv header;

-- routes

-- drop table ptv.routes;

create table ptv.routes(
		route_id varchar,
```

```
            agency_id numeric,
            route_short_name varchar,
            route_long_name varchar,
            route_type numeric,
            route_color varchar,
            route_text_color varchar
            );

copy ptv.routes(route_id, agency_id, route_short_name, route_long_name, route_type, route_color,
route_text_color)
from '/data/adata/gtfs/routes.txt'
delimiter ',' csv header;

-- trips
-- drop table ptv.trips;

create table ptv.trips(
            route_id varchar,
            service_id varchar,
            trip_id varchar,
            shape_id varchar,
            trip_headsign varchar,
            direction_id numeric
            );

copy ptv.trips(route_id, service_id, trip_id, shape_id, trip_headsign, direction_id)
from '/data/adata/gtfs/trips.txt'
delimiter ',' csv header;

    -- shapes
    -- drop table ptv.shapes;

    create table ptv.shapes(
    shape_id varchar,
    shape_pt_lat numeric,
    shape_pt_lon numeric,
    shape_pt_sequence numeric,
    shape_dist_traveled numeric
    );

    copy ptv.shapes(
    shape_id,
    shape_pt_lat,
    shape_pt_lon,
    shape_pt_sequence,
    shape_dist_traveled)
    from
    '/data/adata/gtfs/shapes.txt'
    delimiter ','
```

```
csv header;

-- stop_times
-- drop table ptv.stop_times;

create table ptv.stop_times (
trip_id varchar,
stop_id varchar,
arrival_time varchar,
departure_time varchar,
stop_sequence numeric,
stop_headsign varchar,
pickup_type varchar,
drop_off_type varchar,
shape_dist_traveled varchar
);

copy ptv.stop_times(
trip_id,
arrival_time,
departure_time,
stop_id,
stop_sequence,
stop_headsign,
pickup_type,
drop_off_type,
shape_dist_traveled)
from
'/data/adata/gtfs/stop_times.txt'
delimiter ','
csv header;

-- stops
-- drop table ptv.stops;

create table ptv.stops (
stop_id varchar,
stop_name varchar,
stop_lat numeric,
stop_lon numeric
);

copy ptv.stops (
stop_id,
stop_name,
stop_lat,
stop_lon)
```

```
from
'/data/adata/gtfs/stops.txt'
delimiter ','
csv header;


--calendar
--drop table ptv.calendar;

create table ptv.calendar(service_id varchar,
monday numeric,
tuesday numeric,
wednesday numeric,
thursday numeric,
friday numeric,
saturday numeric,
sunday numeric,
start_date date,
end_date date);

copy ptv.calendar(service_id,
monday,
tuesday,
wednesday,
thursday,
friday,
saturday,
sunday,
start_date,
end_date)
from
'/data/adata/gtfs/calendar.txt'
delimiter ','
csv header;


--calendar_dates
--drop table ptv.calendar_dates;

create table ptv.calendar_dates(service_id varchar,
date date,
exception_type varchar);

copy ptv.calendar_dates(service_id,
date,
exception_type)
from
```

```
'/data/adata/gtfs/calendar_dates.txt'
delimiter ','
csv header;
```

1.3 ABS Mesh Blocks

Scripts to restore the Mesh Blocks files by using correct dataset file. Restore the file using ogr2ogr into table "**mb2021**"

```
ogr2ogr PG:"dbname=gisdb user=postgres"
"/data/adata/MB_2021_AUST_SHP_GDA2020/MB_2021_AUST_GDA2020.shp" -nln ptv.mb2021 -
overwrite -nlt MULTIPOLYGON
```

1.4 ABS Allocation Files

Write the SQL script to restore the LGA2021 Allocation file.

```
 --drop table ptv.lga2021

create table ptv.lga2021(mb_code_2021 varchar,
lga_code_2021 varchar,
lga_name_2021 varchar,
state_code_2021 varchar,
state_name_2021 varchar,
aus_code_2021 varchar,
aus_name_2021 varchar,
area_albers_sqkm numeric,
asgs_loci_uri_2021 varchar
);

copy ptv.lga2021(mb_code_2021,
lga_code_2021,
lga_name_2021,
state_code_2021,
state_name_2021,
aus_code_2021,
aus_name_2021,
area_albers_sqkm,
asgs_loci_uri_2021)
from
'/data/adata/LGA_2021_AUST.csv'
delimiter ','
csv header;
```

Write the SQL script to restore the SAL 2021 Allocation file for suburb2021.

```
-- drop table ptv.suburb2021;
```

```
create table ptv.suburb2021 (
mb_code_2021 varchar,
sal_code_2021 varchar,
sal_name_2021 varchar,
state_code_2021 varchar,
state_name_2021 varchar,
aus_code_2021 varchar,
aus_name_2021 varchar,
area_albers_sqkm numeric,
asgs_loci_uri_2021 varchar
);

copy ptv.suburb2021 (mb_code_2021,
sal_code_2021,
sal_name_2021,
state_code_2021,
state_name_2021,
aus_code_2021,
aus_name_2021,
area_albers_sqkm,
asgs_loci_uri_2021)
from
'/data/adata/SAL_2021_AUST.csv'
delimiter ','
csv header;
```

The allocation tables have 1-N relationship with the mb2021 in mb_code21 – mb_code_2021. Although there are no PK – FK defined in the table, the relationship rule still apply.

1.5 Data Verification

Verify your restoration by running this script. 1.Do not modify the verification script. 2.Make sure that the table name is consistent with the table we provided.

Output: Attach a screenshot of the results to include all tables you have restored in Task 1.

```
with tbl as
(select table_schema, TABLE_NAME
 from information_schema.tables
 where table_schema in ('ptv'))
select table_schema, TABLE_NAME,
(xpath('/row/c/text()', query_to_xml(format('select count(*) as c from %I.%I', table_schema, TABLE_NAME),
FALSE, TRUE, ''))[1]::text::int AS rows_n
from tbl
order by table_name;
```

Screenshot:

	table_schema	table_name	rows_n
1	ptv	agency	10
2	ptv	calendar	380
3	ptv	calendar_dates	15
4	ptv	lga2021	368,286
5	ptv	mb2021	368,286
6	ptv	routes	3,300
7	ptv	shapes	9,757,418
8	ptv	stop_times	8,122,810
9	ptv	stops	27,821
10	ptv	suburb2021	368,286
11	ptv	trips	236,613

Task 2: Data Preprocessing

2.1 Filter Melbourne Metropolitan area

The mb2021 table contains whole mesh blocks in Australia. To minimise the query cost, we want to ensure that you only use the mesh blocks in Melbourne Metropolitan. The Melbourne Metropolitan's mesh blocks can be identified from the gcc_name21. If the column contains "Greater Melbourne", this mesh block is located in Melbourne Metropolitan.

Create a table named "**mb2021_mel**" that contains ONLY the mesh blocks in Melbourne Metropolitan.

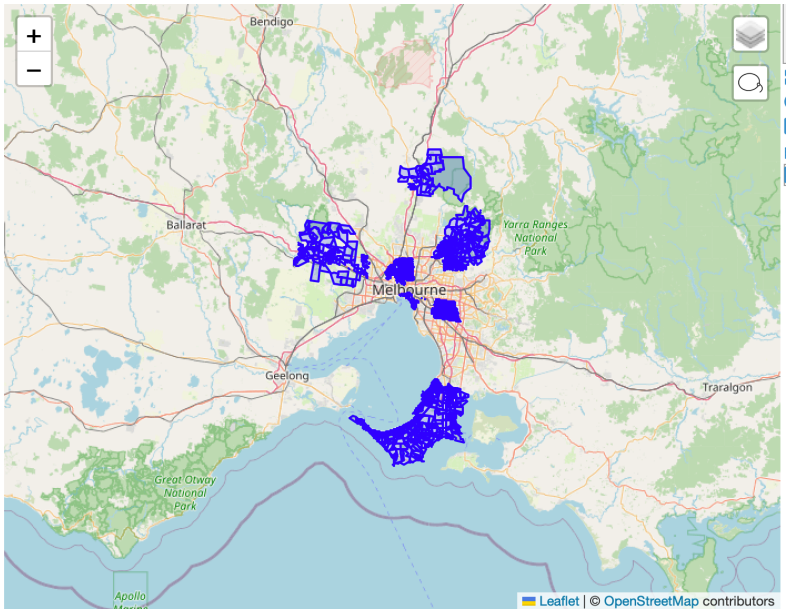
	sa1_code21	sa2_code21	sa2_name21	sa3_code21	sa3_name21	sa4_code21	sa4_name21	gcc_code21	gcc_name21	ste_code21	ste_name21	
22	10901117322	109011173	Albury - North	10901	Albury	109	Murray	1RNSW	Rest of NSW	1	New South Wales	AU
23	10901117322	109011173	Albury - North	10901	Albury	109	Murray	1RNSW	Rest of NSW	1	New South Wales	AU
24	21401137143	214011371	Frankston	21401	Frankston	214	Mornington Peninsula	2GMEL	Greater Melbourne	2	Victoria	AU
25	10901117325	109011173	Albury - North	10901	Albury	109	Murray	1RNSW	Rest of NSW	1	New South Wales	AU
26	10901117301	109011173	Albury - North	10901	Albury	109	Murray	1RNSW	Rest of NSW	1	New South Wales	AU
27	10901117323	109011173	Albury - North	10901	Albury	109	Murray	1RNSW	Rest of NSW	1	New South Wales	AU

Write the SQL script to do this.

```
create table ptv.mb2021_mel as
select
        *
from
        ptv.mb2021 m
where
        upper(m.gcc_name21) = upper('Greater Melbourne');
```

Attach a screenshot of the Spatial Map results.

Screenshot:



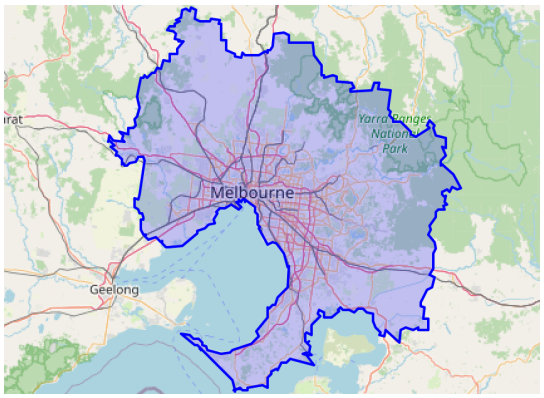
2.2 Melbourne Metropolitan Boundary

Since the working area will be Melbourne Metropolitan, it is important to have a polygon for the boundary of our working area. Create a table, named "**melbourne**" for Melbourne Metropolitan boundary. Hint: aggregate all mesh blocks polygon to create one large polygon for Melbourne Metropolitan boundary.

Write the SQL script to do this.

```
create table ptv.melbourne as
select
        st_union(wkb_geometry) boundary
from
        ptv.mb2021_mel;
```

Attach a screenshot of the Spatial Map results.



2.3 Add Geometry column to Stops table

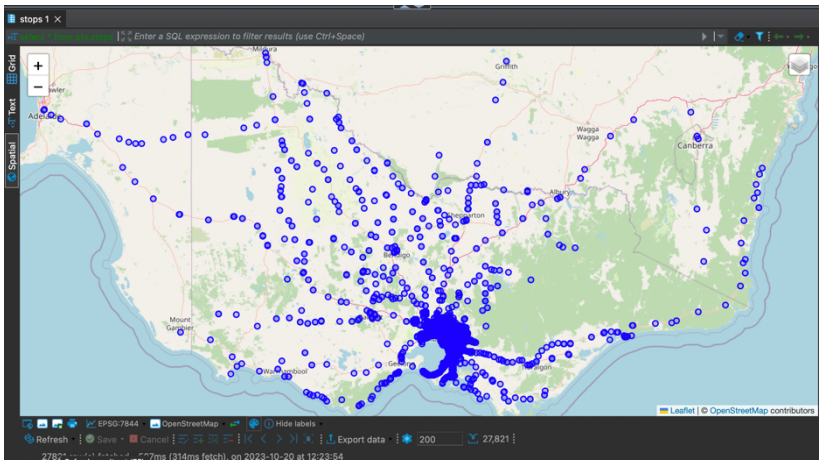
Stops table does not have any geometry column. Add a geometry column by using the latitude and longitude value that are available in the table. Make sure you use GDA2020 (SRID:7844) for this column.

Write the SQL script to do this.

```
select
        *
from
        ptv.stops;

alter table ptv.stops add column geom geometry;

update
        ptv.stops
set
        geom = st_setsrid(st_point(stop_lon,
        stop_lat),
        7844)
where
```

```
        geom is null;

select
        *
from
        ptv.stops;
```

Attach a screenshot of the Spatial Map results.



2.4 Denormalise GTFS structure

The ptv.stops table does not show direct information regarding the vehicle types, routes_short_name and routes_long_name. These information are stored in the routes table.

Create a table called "stops_routes_mel" to encompass the following attributes: stop_id, stop_name, coordinates, route number (derived from routes_short_name), route name (derived from routes_long_name), and vehicle type. This data set should encompass all stops within the Melbourne Metropolitan area.

The vehicle type is determined by the corresponding route type, where:

- 0 corresponds to tram
- 2 corresponds to train
- 3 corresponds to bus
- Any other route type is labeled as 'Unknown'.

Use this figure as an example of expected result. (Note:Data value is for demonstration purposes only.)

	stop_id	stop_name	geom	route_number	route_name	vehicle
1	1000	Dole Ave/Cheddar Rd (Reservoir)	POINT (145.018951051008 -37.7007748061772)	556	Northland SC - Epping Plaza SC	Bus
2	10001	Rex St/Taylors Rd (Kings Park)	POINT (144.776152425766 -37.7269752097248)	418	St Albans Station - Caroline Springs	Bus
3	10002	Yuille St/Centenary Ave (Melton)	POINT (144.595789405033 -37.6761595024019)	458	Melton Station - Kurunjang	Bus
4	10002	Yuille St/Centenary Ave (Melton)	POINT (144.595789405033 -37.6761595024019)	943	Watergardens Station - Melton	Bus
5	10009	Gum Rd/Main Rd West (Albanvale)	POINT (144.775899388911 -37.7414971143014)	424	St Albans Station - Brimbank Central SC	Bus
6	1001	Lloyd Ave/Cheddar Rd (Reservoir)	POINT (145.019685286526 -37.6991830099504)	556	Northland SC - Epping Plaza SC	Bus
7	10010	Kings Rd/Main Rd West (St Albans)	POINT (144.78008474429 -37.7419455261211)	424	St Albans Station - Brimbank Central SC	Bus
8	10011	Moffat St/Main Rd West (St Albans)	POINT (144.783466504334 -37.7423246041254)	424	St Albans Station - Brimbank Central SC	Bus
9	10012	Washington St/Main Rd West (St Albans)	POINT (144.787912291551 -37.7427956612577)	424	St Albans Station - Brimbank Central SC	Bus
10	10013	Kate St/Main Rd West (St Albans)	POINT (144.79457341719 -37.7435693788456)	424	St Albans Station - Brimbank Central SC	Bus
11	10013	Kate St/Main Rd West (St Albans)	POINT (144.79457341719 -37.7435693788456)	425	St Albans Station - Watergardens Station	Bus
12	10014	Raleighs Rd/Centenary Ave (Melton)	POINT (144.588776428553 -37.6753043554238)	458	Melton Station - Kurunjang	Bus

Make sure you remove any duplications in your result.

Write your SQL query here

```
create table ptv.stops_route_mel as (
select
        distinct s.stop_id,
        s.stop_name,
        s.geom as coordinates,
        r.route_short_name as route_number,
        r.route_long_name as route_name,
        case
                when r.route_type = 0 then 'tram'
                when r.route_type = 2 then 'train'
                when r.route_type = 3 then 'bus'
                else 'Unknown'
        end as vehicle_type
from
        ptv.stops s
join
   ptv.stop_times st on
        s.stop_id = st.stop_id
join
   ptv.trips t on
```

```
        st.trip_id = t.trip_id
join
   ptv.routes r on
        t.route_id = r.route_id
where
        ST_Within(s.geom,
        (
        select
                boundary
        from
                ptv.melbourne)));
```

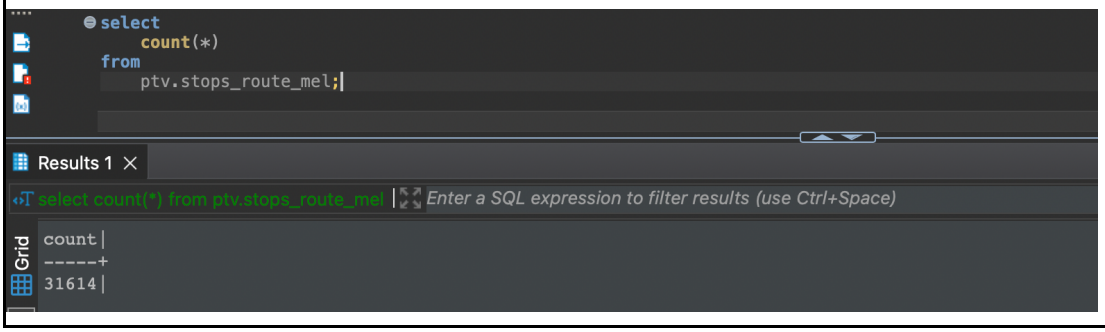
Please complete the following statistics from **stops_routes_mel** table:

Question 2.4.1:How many rows do you have in the stops_routes_mel table?

Write the SQL script to do this and attach a screenshot of the query

```
select
        count(*)
from
        ptv.stops_route_mel;
```

Screenshot

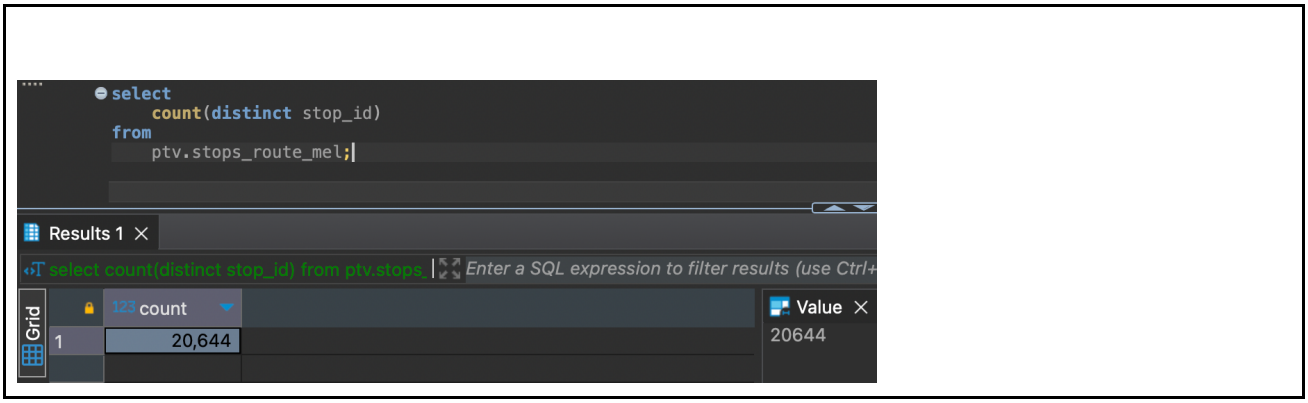


Question 2.4.2:How many unique stop_ids do you have in the stops_routes_mel table?

Write the SQL script to do this and attach a screenshot of the query

```
select
        count(distinct stop_id)
from
        ptv.stops_route_mel;
```

Screenshot



Task 3: Data Analytics

3.1 Suburbs Accessibility

Create an SQL query to identify the **number of bus stops** in each Suburb. Your result should have the suburb name and the total bus stops in it.

Hint :

- identify the mesh block location of a bus stop. Then, aggregate the number in Suburb level.
- One suburb consists of multiple mesh blocks

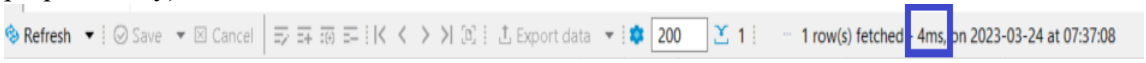
Write your SQL query here

```
with suburb as (
select
        sal_name_2021 suburb_name,
        st_union(wkb_geometry) geom
from
        ptv.suburb2021 s
join ptv.mb2021_mel mm on
        s.mb_code_2021 = mm.mb_code21
group by
        sal_name_2021)
select
        su.suburb_name,
        count(distinct srm.stop_id) num_stops
from
        suburb su
join ptv.stops_route_mel srm
on
        st_contains(su.geom,
        srm.coordinates)
where
        lower(srm.vehicle_type) = lower('bus')
group by
        su.suburb_name;
```

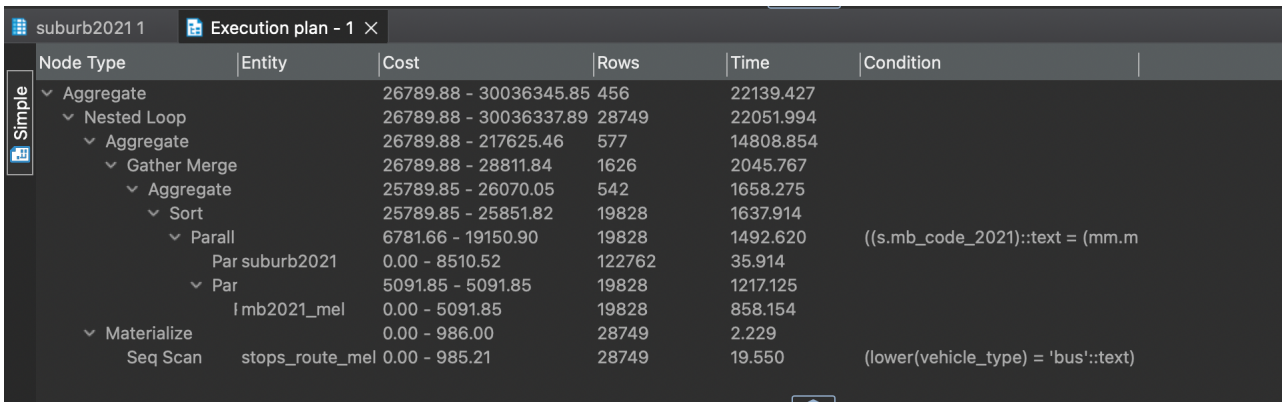
Provide the screenshot of your **execution plan** and the real **execution time** for this query. You can get the real execution time under your result set.

Note:

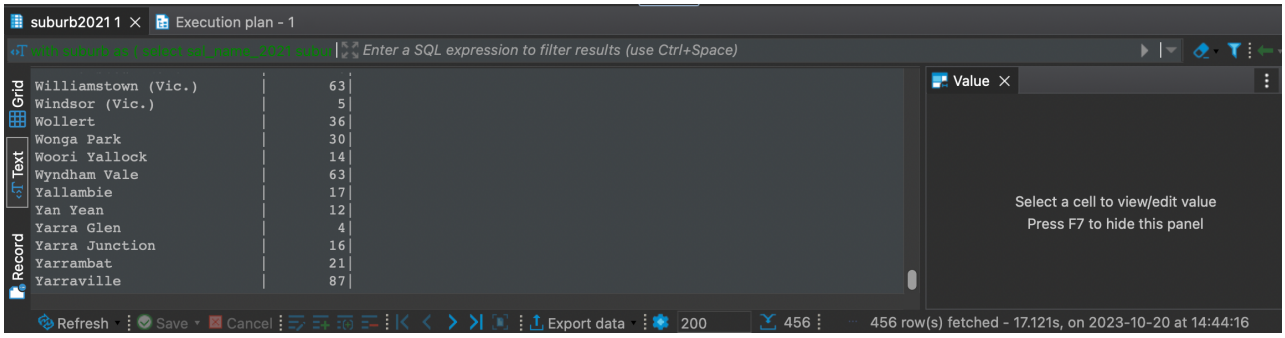
- Execution plan can be found in SQL Editor -> Explain Execution Plan
- The execution time is shown in the screenshot below.(Note: the screenshot is for demonstration purposes only)



Execution plan :



Execution time :



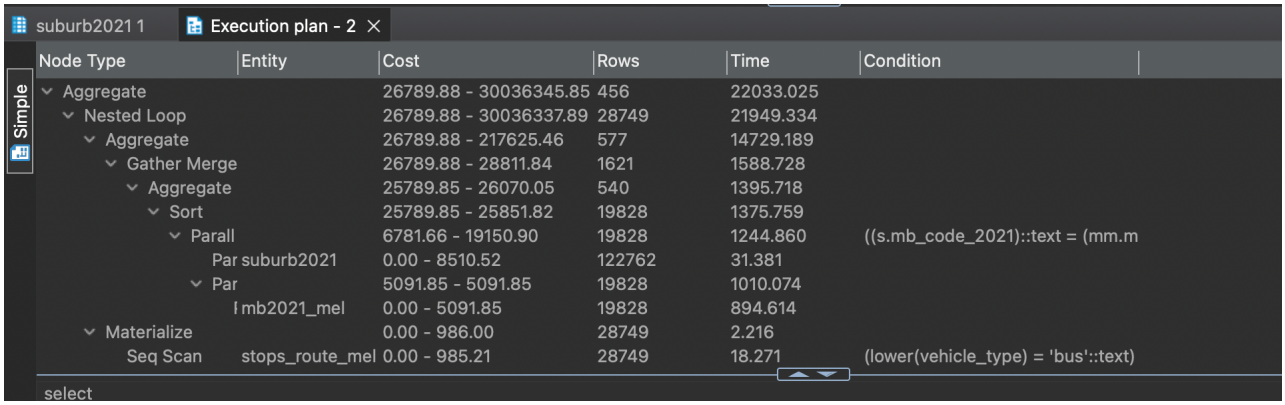
You are now tasked with devising an approach to enhance your query execution and minimize execution time. Provide a comprehensive explanation of your strategy, accompanied by the SQL script outlining the measures you've taken to optimize query performance. Additionally, include a screenshot showcasing the execution plan and execution time, effectively visualizing the enhancements achieved following the optimizatio.

Our strategy is to reduce aggregation and minimise execution time.

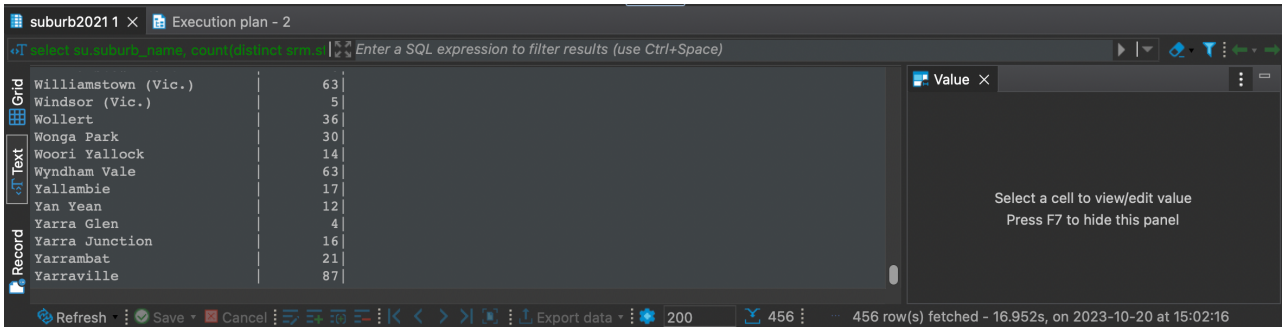
```
select
        su.suburb_name,
        count(distinct srm.stop_id) num_stops
from
        (
select
```

```
        sal_name_2021 suburb_name,
        st_union(wkb_geometry) geom
from
        ptv.suburb2021 s
join ptv.mb2021_mel mm on
        s.mb_code_2021 = mm.mb_code21
group by
        sal_name_2021) su
join ptv.stops_route_mel srm
on
        st_contains(su.geom,
        srm.coordinates)
where
        lower(srm.vehicle_type) = lower('bus')
group by
        su.suburb_name;
```

Improved Execution plan :



Improved Execution Time :



Question 3.1.1: Provide a list of the five suburbs with the lowest count of stops. In case multiple suburbs share the same minimum number of stops in your findings, arrange them in ascending order based on their suburb names.

Write the SQL script to do this and attach a screenshot of the query

```
select
	su.suburb_name,
	count(distinct srm.stop_id) num_stops
from
	(
	select
		sal_name_2021 suburb_name,
		st_union(wkb_geometry) geom
	from
		ptv.suburb2021 s
	join ptv.mb2021_mel mm on
		s.mb_code_2021 = mm.mb_code21
	group by
		sal_name_2021) su
join ptv.stops_route_mel srm
on
	st_contains(su.geom,
	srm.coordinates)
where
	lower(srm.vehicle_type) = lower('bus')
group by
	su.suburb_name
order by
	num_stops,
	suburb_name
limit 5;
```

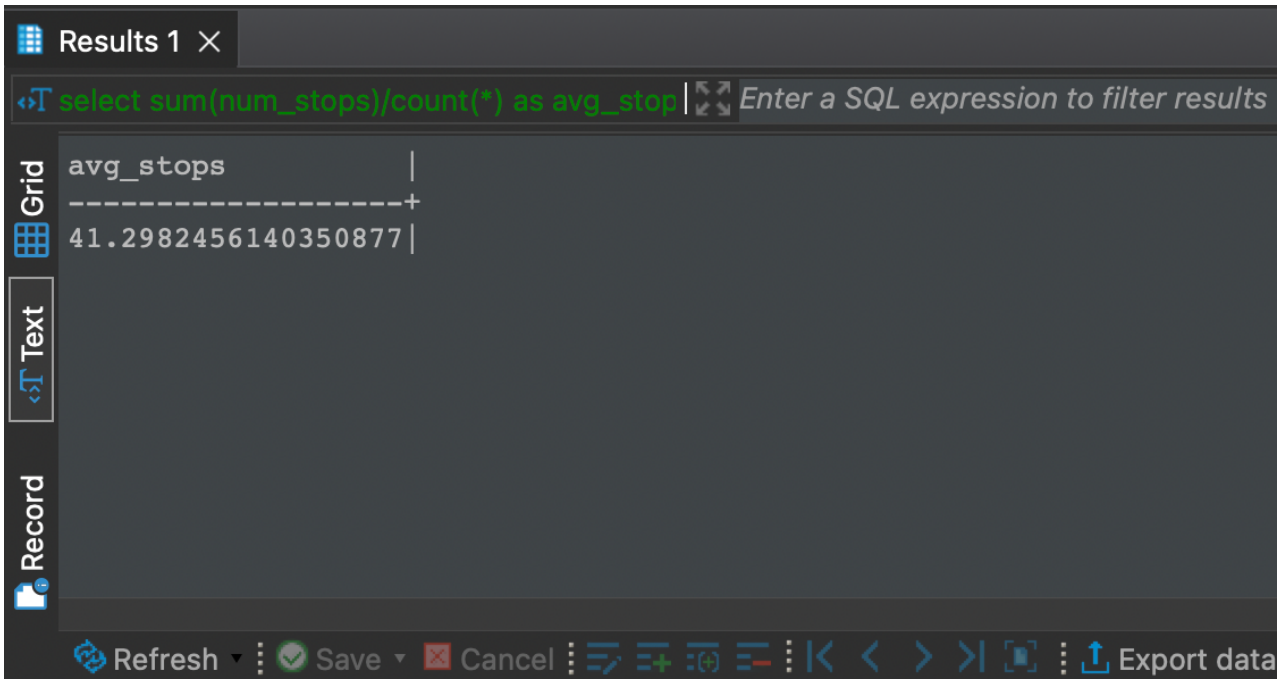
Screenshot

Question 3.1.2: Average number of distinct stops in suburb

Write the SQL script to do this and attach a screenshot of the query

```
select
	sum(num_stops)/ count(*) as avg_stops
from
	(
	select
		su.suburb_name,
		count(distinct srm.stop_id) num_stops
	from
		(
		select
			sal_name_2021 suburb_name,
			st_union(wkb_geometry) geom
		from
			ptv.suburb2021 s
		join ptv.mb2021_mel mm on
			s.mb_code_2021 = mm.mb_code21
		group by
			sal_name_2021) su
	join ptv.stops_route_mel srm
on
		st_contains(su.geom,
		srm.coordinates)
	where
		lower(srm.vehicle_type) = lower('bus')
	group by
		su.suburb_name) as busstops;
```

Screenshot



3.2 LGA Blankspot

The next step is to evaluate the **residential area without Bus Stops**. The residential area without any Bus Stops in it is considered as the **Blankspot**. Each mesh block has a distinct category. The category is defined in "**mb_cat21**" column, mb2021_mel table. In this task, your duty is to identified the percentage of blankspot in every city council or LGA. Below is the blankspot example in Kingsbury.

Let B as the number of residential blankspot, R as the total number of Residential Mesh Blocks, where B ⊆ R. The percentage of blankspot X in every LGA can be calculated using the following formula

$$X = \frac{\sum B}{\sum R} * 100\%$$

Display the LGA name, total number of Residential Mesh Blocks, total number of residential blankspot, percentage of blankspot in Melbourne Metropolitan. Sort results in ascending order by total number of Residential Mesh Blocks.

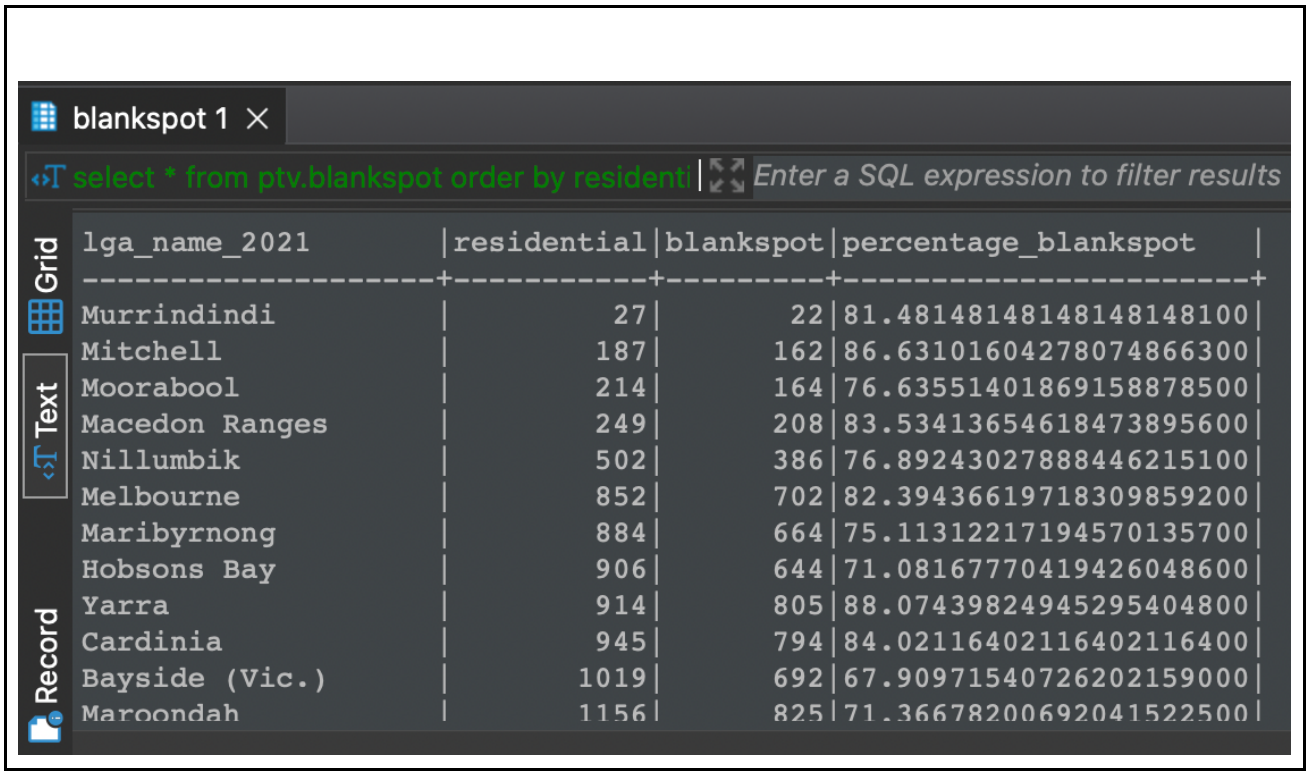
Write the SQL script to do this and attach a screenshot of the query

```
create table ptv.blankspot as (with res as (
select
        mb_code21,
        lga_name_2021,
        st_union(wkb_geometry) geom
from
        ptv.mb2021_mel mm
join ptv.lga2021 l on
        (l.mb_code_2021 = mm.mb_code21)
```

```
where
	lower(mm.mb_cat21) = lower('residential')
group by
	mb_code21,
	lga_name_2021)
select
	lga_name_2021 ,
	(
	select
		residential
	from
		(
		select
			lga_name_2021 lganame,
			count(*) residential
		from
			res
		group by
			lga_name_2021) lga
	where
		lga_name_2021 = lganame) as residential,
	count(*) as blankspot
from
	res
where
	mb_code21 not in (
	select
		mb_code21
	from
		res,
		ptv.stops_route_mel srm
	where
		st_within(srm.coordinates,
		res.geom))
group by
	lga_name_2021);

select *
from
ptv.blankspot
order by residential;

alter table ptv.blankspot add column percentage_blankspot numeric;

update
	ptv.blankspot
set
	percentage_blankspot = cast(blankspot as DECIMAL(7,
	2))/ cast(residential as DECIMAL(7,
	2))* 100
```

```
where
        percentage_blankspot is null;
```

Screenshot



lga_name_2021	residential	blankspot	percentage_blankspot
Murrindindi	27	22	81.48148148148148100
Mitchell	187	162	86.63101604278074866300
Moorabool	214	164	76.63551401869158878500
Macedon Ranges	249	208	83.53413654618473895600
Nillumbik	502	386	76.89243027888446215100
Melbourne	852	702	82.39436619718309859200
Maribyrnong	884	664	75.11312217194570135700
Hobsons Bay	906	644	71.08167770419426048600
Yarra	914	805	88.07439824945295404800
Cardinia	945	794	84.02116402116402116400
Bayside (Vic.)	1019	692	67.90971540726202159000
Maroondah	1156	825	71.36678200692041522500

Question 3.2.1: Complete the following statistical data based on the result.

Note:

- The query and screenshot are not required for this section. You can write down your results directly
- If more than one suburb has the same percentage of blankspots, sort them by suburb name in ascending order.

Criteria	Answer
Top 5 LGAs with the highest % of blankspots	Yarra Mitchell Cardinia Macedon Ranges Port Phillip
Top 5 LGAs with the lowest % of blankspots	Manningham Whitehorse Bayside (Vic.)

	Knox Darebin
Average % of blankspots	75.28204349813989700311

Task 4: Data Visualisation

In this task, you are required to incorporate a heatmap visualization.

4.1 LGA Blankspot Analysis

In this task, you will put the blankspot percentage in the heatmap. Provide the segmentation as follow:

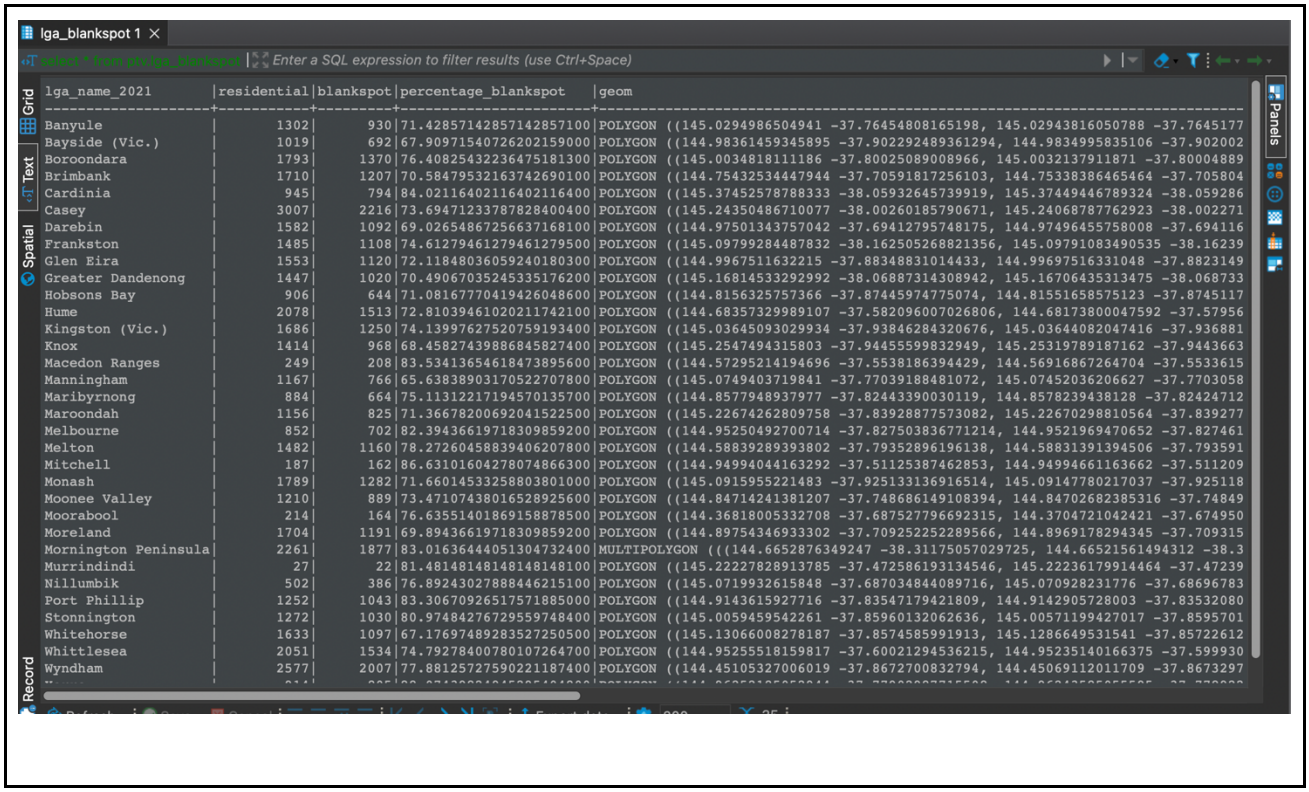
- $X \leq 20\ \%$
- $20\% < X \leq 40\%$
- $40\% < X \leq 60\%$
- $60\% < X \leq 80\%$
- $X > 80\%$

Write an SQL query to create a table named 'lga_blankspot' containing the blankspot percentages categorised by the LGA from the previous task 3.2 and sufficient spatial data. And attach a screenshot of the table contents.

Note: Ensure that this table is structured to facilitate the creation of a visual heat map specifically for the Melbourne region.

```
create table ptv.lga_blankspot as (
with res as (
select
        lga_name_2021 lganame,
        st_union(wkb_geometry) geom
from
        ptv.mb2021_mel mm
join ptv.lga2021 l on
        (l.mb_code_2021 = mm.mb_code21)
group by
        lga_name_2021)
select
        b.lga_name_2021 ,
        b.residential ,
        b.blankspot ,
        b.percentage_blankspot,
        r.geom
from
        ptv.blankspot b
join res r on
        (b.lga_name_2021 = r.lganame));
```

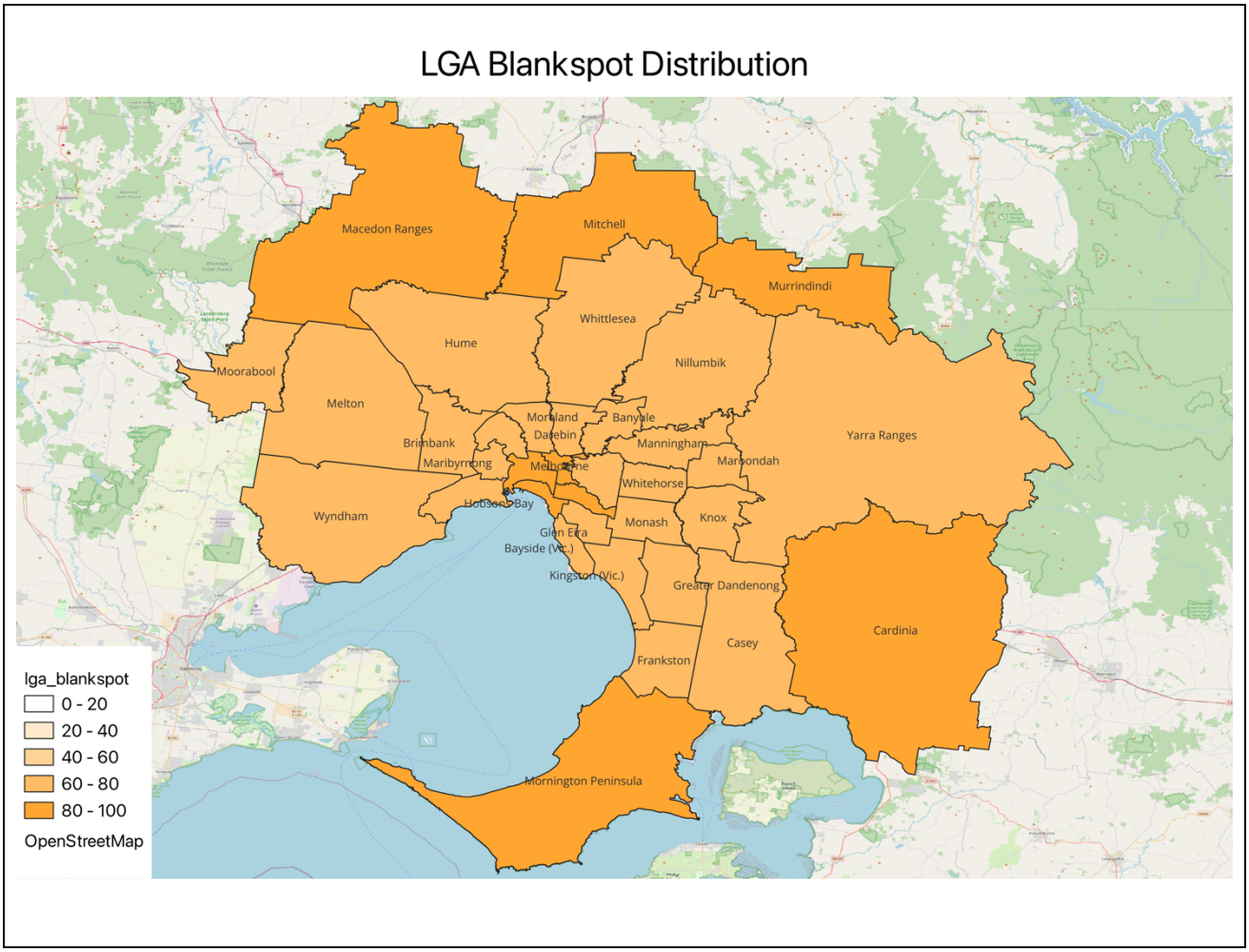
Screenshot



lga_name_2021	residential	blankspot	percentage_blankspot	geom
Banyule	1302	930	71.42857142857142857100	POLYGON ((145.0294986504941 -37.76454808165198, 145.02943816050788 -37.7645177
Bayside (Vic.)	1019	692	67.90971540726202159000	POLYGON ((144.98361459345895 -37.902292489361294, 144.9834995835106 -37.902002
Boroondara	1793	1370	76.40825432236475181300	POLYGON ((145.0034818111186 -37.80025089008966, 145.0032137911871 -37.80004889
Brimbank	1710	1207	70.58479532163742690100	POLYGON ((144.75432534447944 -37.70591817256103, 144.75338386465464 -37.705804
Cardinia	945	794	84.02116402116402116400	POLYGON ((145.37452578788333 -38.05932645739919, 145.37449446789324 -38.059286
Casey	3007	2216	73.69471233787828400400	POLYGON ((145.24350486710077 -38.00260185790671, 145.24068787762923 -38.002271
Darebin	1582	1092	69.02654867256637168100	POLYGON ((144.97501343757042 -37.69412795748175, 144.97496455758008 -37.694116
Frankston	1485	1108	74.61279461279461279500	POLYGON ((145.09799284487832 -38.162505268821356, 145.09791083490535 -38.16239
Glen Eira	1553	1120	72.11848036059240180300	POLYGON ((144.9967511632215 -37.88348831014433, 144.99697516331048 -37.8823149
Greater Dandenong	1447	1020	70.49067035245335176200	POLYGON ((145.16814533292992 -38.06887314308942, 145.16706435313475 -38.068733
Hobsons Bay	906	644	71.08167770419426048600	POLYGON ((144.8156325757366 -37.87445974775074, 144.81551658575123 -37.8745117
Hume	2078	1513	72.81039461020211742100	POLYGON ((144.68357329989107 -37.582096007026806, 144.68173800047592 -37.57956
Kingston (Vic.)	1686	1250	74.13997627520759193400	POLYGON ((145.03645093029934 -37.93846284320676, 145.03644082047416 -37.936881
Knox	1414	968	68.45827439886845827400	POLYGON ((145.2547494315803 -37.94455599832949, 145.25319789187162 -37.9443663
Macedon Ranges	249	208	83.53413654618473895600	POLYGON ((144.57295214194696 -37.5538186394429, 144.56916867264704 -37.5533615
Manningham	1167	766	65.63838903170522707800	POLYGON ((145.0749403719841 -37.77039188481072, 145.07452036206627 -37.7703058
Maribyrnong	884	664	75.11312217194570135700	POLYGON ((144.8577948937977 -37.82443390030119, 144.8578239438128 -37.82424712
Maroondah	1156	825	71.36678200692041522500	POLYGON ((145.22674262809758 -37.83928877573082, 145.22670298810564 -37.839277
Melbourne	852	702	82.39436619718309859200	POLYGON ((144.95250492700714 -37.827503836771214, 144.9521969470652 -37.827461
Melton	1482	1160	78.27260458839406207800	POLYGON ((144.58839289393802 -37.79352896196138, 144.58831391394506 -37.793591
Mitchell	187	162	86.63101604278074866300	POLYGON ((144.94994044163292 -37.51125387462853, 144.94994661163662 -37.511209
Monash	1789	1282	71.66014533258803801000	POLYGON ((145.0915955221483 -37.925133136916514, 145.09147780217037 -37.925118
Moonee Valley	1210	889	73.47107438016528925600	POLYGON ((144.84714241381207 -37.748686149108394, 144.84702682385316 -37.74849
Moorabool	214	164	76.63551401869158878500	POLYGON ((144.36818005332708 -37.687527796692315, 144.3704721042421 -37.674950
Moreland	1704	1191	69.89436619718309859200	POLYGON ((144.89754346933302 -37.709252252289566, 144.8969178294345 -37.709315
Mornington Peninsula	2261	1877	83.01636444051304732400	MULTIPOLYGON (((144.6652876349247 -38.31175057029725, 144.66521561494312 -38.3
Murrindindi	27	22	81.48148148148148148100	POLYGON ((145.22227828913785 -37.472586193134546, 145.22236179914464 -37.47239
Nillumbik	502	386	76.89243027888446215100	POLYGON ((145.0719932615848 -37.687034844089716, 145.070928231776 -37.68696783
Port Phillip	1252	1043	83.30670926517571885000	POLYGON ((144.9143615927716 -37.83547179421809, 144.9142905728003 -37.83532080
Stonnington	1272	1030	80.97484276729559748400	POLYGON ((145.0059459542261 -37.85960132062636, 145.00571199427017 -37.8595701
Whitehorse	1633	1097	67.17697489283527250500	POLYGON ((145.13066008278187 -37.8574585991913, 145.1286649531541 -37.85722612
Whittlesea	2051	1534	74.79278400780107264700	POLYGON ((144.95255518159817 -37.60021294536215, 144.95235140166375 -37.599930
Wyndham	2577	2007	77.88125727590221187400	POLYGON ((144.45105327006019 -37.8672700832794, 144.45069112011709 -37.8673297

Provide the screenshot for the heatmap by using QGIS. Please note that the heatmap is map-based and visually represents the distribution of blankspot percentages across different LGAs in the Melbourne region.

Remember to include appropriate labels, titles, and legends in your visualizations to make them easy to understand (Updated 05/10/2023)


LGA Blankspot Distribution
Macedon Ranges
Mitchell
Murrindindi
Whittlesea
Hume
Nillumbik
Moorabool
Melton
Moreland
Banyule
Darebin
Manningham
Yarra Ranges
Brimbank
Maribyrnong
Melbourne
Maroondah
Whitehorse
Wyndham
Hobsons Bay
Monash
Knox
Glen Eira
Bayside (Vic.)
Kingston (Vic.)
Greater Dandenong
Cardinia
Casey
Frankston
Mornington Peninsula
lga_blankspot
0 - 20
20 - 40
40 - 60
60 - 80
80 - 100
OpenStreetMap

End