GEOM90008 Spatial Data Management Group Assignment

Drone Management Database For Universities

Group 22

Jiaming XIAN 1336110

jiamxian@student.unimelb.edu.au

Sijia Pei 1227269

sipei@student.unimelb.edu.au

XINGTING ZONG

xingtingz@student.unimelb.edu.au

YANXI Ke 1288060

yake@student.unimelb.edu.au

Contents

1. Project Introduction

The University of Melbourne is currently planning to use drones to advance space science research, and as such, a database (DB) based system is required to manage all data related to drones. The University needs this system to help operators coordinate flight plans to ensure flight paths comply with relevant regulations and to keep track of the operational status of the drones for compliance.

The database system manages the drone route data in terms of origin and destination, altitude, and altitude of the flight. This data can be easily tracked, queried, and modified to ensure the proper operation of the drone.

It also could store and update regulatory rules such as CASA regulations. The system is updated and modified in real-time to ensure that the rules for regulation are up to date with the latest standards. The battery and sensor status of the drone can be monitored in real-time with relevant information stored to ensure safe flight and to alert managers to replace hardware. The system can record flight data and analyze the flight history data for optimization and improvement. The advantage of the DB system is the visualization of the spatial data, which makes the user's reference and access to the data clear and straightforward. For this, we used QGIS and PgAdmin to build and manage the database system for our visualization, and used data from the following data sources as data for our database visualization:

- Crowdsourced Data: OpenStreetMap (OSM).
- Authoritative Source Data: VicMap, Unimelb Open Spatial Data, Digital boundary files from the Australian Bureau of Statistics.
- User-provided Data: flight path, drone, battery, sensor.

The main purpose of the system is to store and manage all data relating to drones. It allows the user to perform flight space calculations and queries based on the data to ensure that flight paths comply with relevant regulations, and to plan drone operations by monitoring drone activity.

The system will proactively confirm that each flight plan complies with a litany of drone flight regulations and extends this to the management of operators and equipment status management. It also provides additional functionality such as locating the nearest hotels and first aid points near the flight path. Ultimately, the DB system will serve as a reliable and versatile platform for the effective and safe management of drones around the University's space research objectives.

2. Implemented Requirements

2.1 In-scope

2.1.1. Flight plan validation

As the core function of the system, the function aims to utilize the flight plan information and spatial data stored in the database to automatically validate the compliance of drone flight plans with the regulations set by the Civil Aviation Safety Authority (CASA) of Australia. By leveraging advanced algorithms and database resources, the system is capable of effectively identifying any deviations or non-compliance with regulatory requirements. The specific

regulatory requirements for the flight plan, specific implementation methods, and analysis are as follows:

- The flight route should stay clear of surrounding public facilities, including airports, beaches, stadiums, parks, and other similar infrastructure. Drones weighing over 250 grams must maintain a distance of at least 5.5 kilometers from controlled airports. However, for helipads and aerodromes without control towers, drones can operate within a 5.5-kilometer radius. Due to the lack of specific control tower data, the system ultimately adopts a rule where flight routes should not approach an airport within a 5.5-kilometer radius.
    - Implement:
      The system utilizes QGIS to generate corresponding dataset(public_places) of public facilities based on Australian geographical data (Australia-latest-free) using SQL. Then, through SQL, the system determines whether the flight route intersects. The specific SQL code and result are in 5.3.1.
    - Analysis:
      The system pre-processes and generates a dedicated database to store public facilities using QGIS. This approach helps avoid repetitive processing and significantly speeds up the overall processing time. Without this step, it would require additional time to open the Australian data, process it, and generate the relevant areas of public facilities. Due to lack of relevant airport data, there is no control tower. The relevant requirement has been modified to restrict any unmanned aerial vehicles from approaching the airport within a radius of 5.5 kilometers.
- The combined weight of sensors and batteries carried by the drone should be less than the payload capacity of the drone.
    - Implement:
      The system uses SQL to determine whether the total “weight” of sensors and batteries carried by the drone is less than the "max_payload" attribute specified for that particular drone in the drone database. The specific SQL code and result are in 5.3.2.
    - Analysis:
      When initially designing the drone table, the "max_payload" attribute was indeed considered based on the scenario where the drone carries one sensor and one battery. This approach aims to facilitate management and minimize potential conflicts and risks. By standardizing the criterion based on battery capacity and flight distance, it reduces the management and legal risks associated with different drones having varying batteries and sensors. This simplifies the decision-making process and ensures consistent compliance with regulations and safety protocols.
- The batteries carried by the drone should have the capacity to complete the distance specified in the drone's flight mission.
    - Implement:
      The system utilizes SQL to determine if the product of the "power" attribute of the drone's carried battery is greater than the specified flight distance. If the result is

greater than the flight distance, the system will reject the flight plan. The specific SQL code and result are in 5.3.3.

- Analysis:
  We conducted research on popular drone brands in the market and gathered the m/mAh data. Here are the results:
  - DJI Mavic 2 Pro: 9.6m/mAh
  - Autel Robotics EVO II: 6.8m/mAh
  - DJI Phantom 4 Pro: 6.1m/mAh
  - DJI Mini 2: 13.2m/mAh
  - Parrot Anafi: 8.5m/mAh
  - DJI Phantom 3 Professional: 4.96m/mAh
  - Parrot Bebop 2: 8.8m/mAh

  Based on the survey and analysis above, considering the variations in external environmental factors (such as wind speed, atmospheric pressure) and drone energy consumption along different flight paths, it has been decided to use the criterion of battery capacity multiplied by 4 being greater than the distance as the determining factor for whether a drone can complete a flight route. This approach allows for easier calculation and statistical analysis.

- The flight route should be within a range of 600 meters from the drone's take-off point, which corresponds to the operator's visual line of sight.
  - Implement:
    The system uses SQL to determine whether the drone's flight route exceeds the 600-meter range from the starting point. The specific SQL code and result are in 5.3.4.
  - Analysis:
    The choice of a 600-meter range is derived from the specifications provided by many drone manufacturers. It is worth noting that the control range of many consumer drones is limited to around 600 or 700 meters. Considering that the visual range of a person is typically greater than 600 meters, using the drone's control range as the maximum distance it can be away from the operator is a reasonable approach.
- The altitude of the flight route should be below 120 meters above ground level, with a minimum ground level of 15 meters.
  - Implement:
    Based on the given flight route, the system calculates the highest point and the lowest point in the flight, which are the altitude with reference to sea level. Then, based on the DEM raster data (stored in spatial. victoria_dem_30m), we calculate the heights of the highest point and the lowest point of the flight above the ground level, and check whether they meet the conditions that the highest point is no more than 120m and the lowest point is no lower than 15m.The specific SQL code is in 5.3.5.
- The flight plan assignment should ensure that an operator is not responsible for multiple tasks simultaneously. This practice increases the likelihood of safety accidents and adds

complexity to the management of flight personnel data, including the potential for falsification of flight time records.

- Implement:
  The system will use SQL to determine If the start time of the new flight plan is not earlier than the latest end time of all flight plans of the pilot, it can be executed, otherwise it cannot. Then this flight plan will be rejected. The specific SQL code and result are in 5.3.6.
- Analysis:
  To determine if an operator is currently on a mission, we can examine their flight records. Instead of creating a separate attribute in the "operator" table to indicate the mission status, we can combine the mission status, start time, and end time to enhance the utilization of attributes. This approach may be slightly more complex, but it provides a more comprehensive way of tracking an operator's mission status.

- For flight plans assigned to student pilots or those undergoing training, the flight routes should be limited to the school premises. This restriction is implemented for safety considerations.
  - Implement:
    The system checks if the operator assigned to this flight mission has the "licence" attribute set to "Y" in the "operator" table using SQL. If it is set to "Y," the flight route should not exceed the boundaries of the University of Melbourne's campus (specified in the "unimelb_campus" table with campus='Parkville'). The specific SQL code and result are in 5.3.7.

2.1.2.Operator Management

The purpose of this functionality is to utilize database querying and design to achieve data analytics and status management for flight personnel. It should fulfil the following requirements:

- The system can identify employee ID, names and employee contact information.
  - Implement:
    We have built an operator database within the project. The system can retrieve detailed information about the required operators using SQL queries. The specific SQL code and result are in 5.3.8.
- The system should record the operator’s refresher courses, identify all operators who need training next month, and calculate the operator’s flight time.
  - Implement:
    The system will perform a SQL check on the "last_train_data" attribute of all operators in the "operator" table to determine if it is more than 1 year ago. If it is less than one year, it will then check the "flight_history" table to see if the flight hours of the pilot in the past year exceed 50 hours. The specific SQL code and result are in 5.3.9.

2.1.3.Device Management

The purpose of this functionality is to utilize database querying and design to achieve data analytics and status management for drones, as well as related sensors and batteries. It should

also facilitate integration with other functionalities to enable seamless coordination. The specific requirements for this functionality are as follows:

- The system should store and show maintenance dates for all drones, sensors, and batteries.
    - Implement:
      The system determines, using SQL queries to show the drone , sensor, battery table. The specific SQL code and result are in 5.3.14.
- The system should show that the maximum inspection interval for all batteries exceeds 2 years, and the inspection interval for all sensors exceeds 1 year.
    - Implement:
      The system determines, using SQL queries, all the entries in the battery table where the manufacture_date exceeds 2 years from the current time, as well as all the entries in the sensor table where the last_calibration_date exceeds 1 year from the current time. The specific SQL code and result are in 5.3.10.

2.1.4.Flight History

- The system should track the operator of flights and record the historical flights supported by each operator.
    - Implement:
      The system displays all the historical data from the flight_history table using SQL. The specific SQL code is in 5.3.11.

2.1.5.Addition Support function

In addition, the system should provide the following helpful features for drone flights:

- The system will provide information on the three closest hotels and one nearest emergency point around the starting point of the flight route.
    - Implement:
      The system uses SQL to extract all emergency data from the Australian map data and create an emergency table. It also extracts all hotel data to create a hotel table. Then, using SQL, it calculates the three closest hotels to the starting point of the flight route and the closest emergency location. The specific SQL code is in 5.3.12.
    - Analysis:
      Generating corresponding hotel and emergency tables can greatly speed up the search time and avoid opening the table that originally stores all Australian buildings.
- The system will calculate and provide the shortest distance from the center of the University of Melbourne to a designated target point in the flight plan.
    - Implement:
      The system utilizes SQL to retrieve the nearest distance from the center point of a school to the flight route. Specifically, it employs the pgr_dijkstra method to calculate the shortest path. The specific SQL code is in 5.3.13.
    - Analysis:
- The system will determine that drone flights should take place during daylight hours.
    - Implement:

The system utilizes SQL to Use SQL to find whether the start_time and end_time of flight_history are within the range of 6am-5pm. The specific SQL code is in 5.3.15.

- Analysis:
  Since Australian time has daylight saving time and wintertime, the relevant data needs more analysis. Here we uniformly define 6am-5pm as daytime. This is convenient for calculation and statistics.

2.2 Out-scope

There are several functionalities that, due to various reasons, cannot be implemented, and thus we consider them out of scope for the project. Here are these specific functionalities and their reasons:

- In CASA, it is explicitly stated that drones should maintain a minimum distance of 30 meters from people.
  - Reason:
    Due to the lack of specific capabilities such as visual recognition, radar, or similar technologies to identify nearby individuals and gather relevant data, implementing this requirement is not feasible for us.
- CASA states that it is necessary to ensure that the drone remains within the operator's visual line of sight.
  - Reason:
    However, factors such as wall obstructions, terrain obstacles, and similar limitations can prevent the drone from staying within the operator's visual line of sight. Due to our current technological capabilities, project timeline, and other factors, implementing this functionality goes beyond our planned scope for the project. Therefore, this functionality is not included within the scope of our project.
- According to CASA regulations, drones are not permitted to fly into clouds or fog.
  - Reason:
    Since we lack the necessary weather data to make informed judgments regarding cloud or fog presence, we are unable to fulfil this requirement. Therefore, we consider this particular requirement as out of scope for our project.
- According to CASA regulations, drones are not allowed to approach or fly over areas that may pose a risk to public safety, such as accident scenes or fires. Drones must not cause harm to private property or endanger the lives of others.
  - Reason:
    We don’t have relevant data. So, we can’t not implement it.
- The flight route should not traverse any buildings to avoid potential legal disputes and unforeseen consequences that may arise from accidents during flight.
  - Reason:
    Due to the lack of data regarding the height of buildings, the determination becomes significantly complex. Additionally, considering factors such as

building height would require addressing the issue of operator visibility. Therefore, we have decided to classify this requirement as out of scope, as it presents challenges without the necessary data and considerations for operator visibility.

- According to CASA regulations, drones are prohibited from capturing images or videos without the consent of individuals involved.
    - Reason:
      Since we lack the necessary capabilities and functionality to obtain such consent or handle the associated data, this particular functionality is not within the scope of our project.

3. Technical Design

The current database is used to manage all information related to drones, and its logic diagram is detailed in the **5.1 Logical E-R diagram**. As shown in the figure, its structure is composed of 12 tables.

- Firstly, there is the user table, which represents the users who use this database. There are three types of users: admin, operator, and student. The user table records their names, types, email addresses, and account passwords.
- Each operator has a user account. The operator table records the ID, name, employee number (only the operator has an employee number), email, phone number, licence (student or operator), last training time, and total training time of each user who can operate the drone. Among them, students do not need training.
- Each user who can operate a drone has an independent drone, so each drone table contains the drone ID, the maximum load of the drone, and the ID of the operator to which it belongs. The maximum load of a drone needs to be ensured to be greater than the weight of the batteries and sensors it is equipped with.
- Each drone has a sensor, but some backup sensors are not being used. The sensor table has the sensor ID, type, weight, maximum height, last calibration date, and the ID of the drone it belongs to (the drone ID of the backup sensor is null)
- Each drone also uses a battery and some backup batteries. The battery table contains the battery ID, model, make, weight, age, power, manufactured date, and the drone ID to which it belongs (the drone ID of the backup battery is null).
- Each user operating the drone has one or more flight plans, flight_history table has fh_ ID, flight path, ID of the operator executing the flight history, whether the flight complies with the rules, start time, end time, total flight time, start position, and end position.
- The public places (including airports) that flight histories need to avoid, and the columns of the emergency and hotel tables that flight histories need to provide are almost the same, each with their own ID, name, fclass, and geom. Each flight history may use duplicate public places, emergencies, and hotels.

The operator table and flight_history table is the most frequently updated table, and the operator table updates the pilot's flight time every day. Flight_history table adds a new flight history for each flight. The sensor and battery tables will update the ID of the drone they belong to when the sensor and battery are damaged or expired. The user table will only be updated when new users are added, or their information needs to be updated. In order to improve update efficiency and ensure data integrity, the database adopts two related technologies to improve database performance.

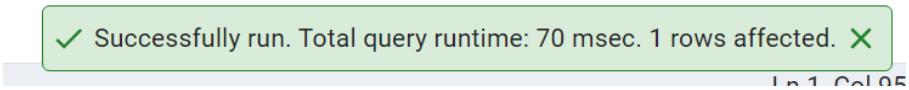
1. Performance: Index
   Indexing can improve data retrieval performance and update efficiency. Indexing can automatically define a data structure that is suitable for the current data table, helping users quickly locate data that meets the conditions. At the same time, it can reduce data access and allow the database to only access index columns, greatly improving retrieval speed. Indexing can also help improve database update efficiency by updating only the data in the index columns, rather than updating all data. When processing ordinary data such as primary keys, no-spatial indexes can be used, while spatial indexes can be used when processing spatial data. The database system in this project requires spatial indexes to help optimize the search performance of geometry data. The specific indexes are shown in the table below:

Table Name	Column Name	Index Type	Index Name
emergency	geom	spatial	emergency_geom_idx
hotel	geom	spatial	hotel_geom_idx
public_places	geom	spatial	public_places_geom_idx
flight_history	startlocation	spatial	flight_history_startlocation_idx
flight_history	endlocation	spatial	flight_history_endlocation_idx
flight_history	path	spatial	flight_history_path_idx
operator	o_id	no-spatial	operator_o_id_idx

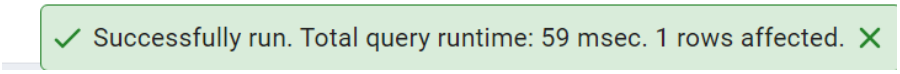
   This is an example of creating indexes:

```
CREATE INDEX hotel_geom_idx
ON group22.hotel
USING GIST (geom);

CREATE INDEX emergency_geom_idx
ON group22.emergency
USING GIST (geom);

CREATE INDEX public_places_geom_idx
ON group22.public_places
USING GIST (geom);
```

Before creating index:



After creating index:

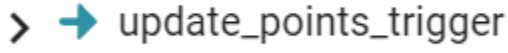


2. Integrity: Trigger

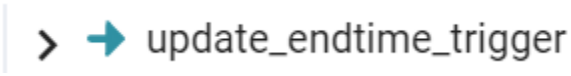
   The database system in this project requires frequent updates, such as updating a flight record after each operator's flight, and updating the operator's flight duration. So, we need a way to ensure the integrity of the data during the update process, and Trigger can help the database achieve this goal. Trigger can ensure that when one of the data in two associated tables is changed, it automatically detects whether the associated data has synchronized changes. If not, Trigger will operate on the associated data according to its defined function to ensure data consistency. It helps administrators better monitor the flight time of operators and more conveniently search for past flight records.

   This database system mainly uses four triggers:

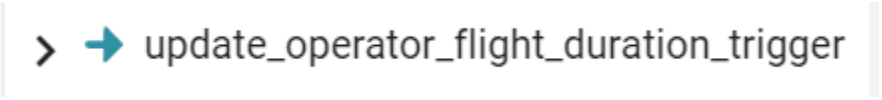
   1. According to provide Path in the flight_history to automatically calculate the starting and ending points.

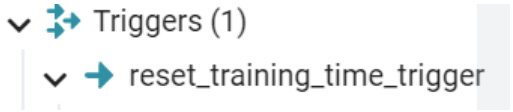
2. According to starttime and flight speed given in the flight_history (based on 5m/s flight speed), automatically calculating the end time and total flight duration (end time start time)



3. According to the total_flight_ time duration calculated in the history automatically updates the corresponding operator's total flight duration in the operator.



4. Automatically reset the total train time of the current stage to zero after each training by the operator



4. Data Sources and Processing

The data sources used in this project can be divided into three categories: crowdsourced data, authoritative source data, and user-provided data. The specific usage is as follows:

- Crowdsourced Data: OpenStreetMap (OSM).
- Authoritative Source Data: VicMap, Unimelb Open Spatial Data, Digital boundary files from Australian Bureau of Statistics.
- User-provided Data: flight path, drone, battery, sensor.

4.1 Crowdsourced Data

The crowdsourced data used in this project comes from OpenStreetMap (OSM), which is mainly used for two purposes: firstly, using Australian regional data of OSM, and secondly, using OSM as a base map in the visualization of QGIS software to provide map background for the project.

The Australian data provided by OSM can be accessed from the following link http://download.geofabrik.de/australia-oceania/australia.html. This file remains updated in real-time, and the OSM data used in this project is up to 2023-05-27T 20:21:39Z. The free shapes contain the name "free" in each layer name to differentiate them and all layers use the "osm_" prefix for their names. Also, all file names contain "_ a_ " means the geometry data included in is MultiPolygon, otherwise it is Point. All coordinates are unprojected WGS84 (EPSG: 4326) and all strings are encoded in UTF-8.

The specific situation of the Geofabrik data used in this system is as follows (filtered by column fclass to obtain the name and geom columns):

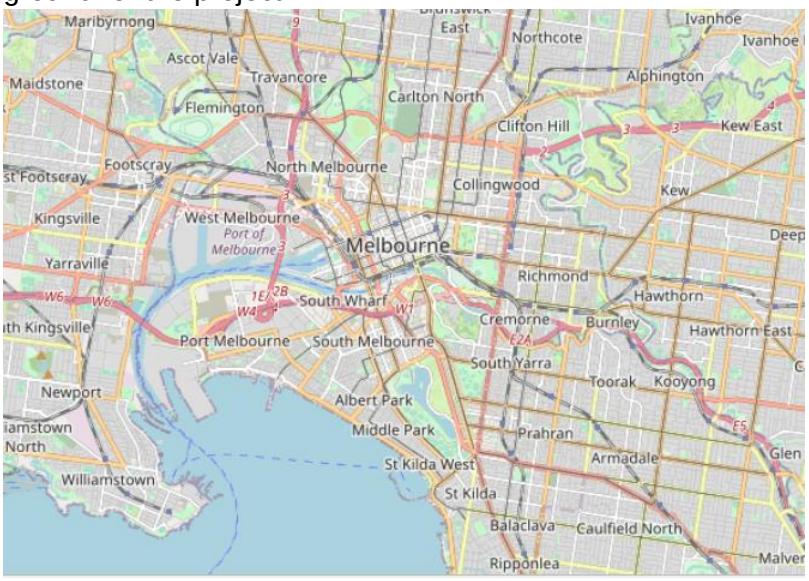
- Points of Interest layer: "hospital", "clinic", "doctors", "sports_centre", "pitch", "stadium", "hotel", "motel", "theme_park".
- Natural Features layer: "beach".
- Transport" layer: "airport", "airfield", "helipad", "apron".

Note: These external data should be stored in the geographic reference system 7855. It is necessary to transform the original data in the geographic reference system 4326 to the geographic reference system 7855 through a one-step reprojection.

Layer	fclass	Geometry (Transformed to EPSG: 7855)	Output Table
Point of Interest	hospital	Point	group22.emergency
	clinic		
	doctors		
	sports_centre	MultiPolygon	group22.public_places
	pitch		
	stadium		
	hotel	Point	group22.hotel
	motel		
	theme_park	MultiPolygon	group22.public_places
Natural Features	beach	MultiPolygon	group22.public_places
Transport	airport	MultiPolygon	group22.public_places
	airfield		
	helipad		
	apron		

Table 4.1 Data Usage of OSM Geofabrik

In addition, the visualization of QGIS software uses OSM as the base map to provide a map background for the project.



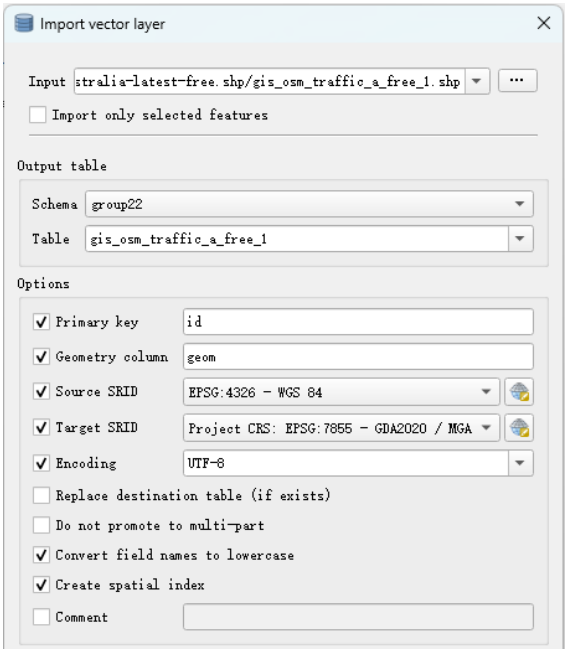
Picture 4.1 OpenStreetMap as the basic map

However, we summarize that open-source databases such as OSM also have some natural shortcomings that may lead to some accuracy issues in our project, as follows:

- Missing data: Some regions may have relatively few or missing data, especially in sparsely populated or geographically remote areas, where the coverage and level of detail of the data may be limited.
- Difficulty in data validation: because the data is edited by a large number of contributors, it will become difficult to verify the accuracy and integrity of the data. For example, with airport data, our team is unable to verify whether each airport and apron is genuine and still in use.
- Data quality issues: There may be differences in data quality, as different users may have different collection methods and standards, resulting in insufficient data consistency and accuracy. For example, the location of certain hotels may be located on the road or at a certain distance from the building. And, we have noticed that the OSM database we use defines all green spaces as parks, rather than tourist destinations for people's entertainment as we understand them. Even the South Lawn of Parkville campus is included in the park layer.

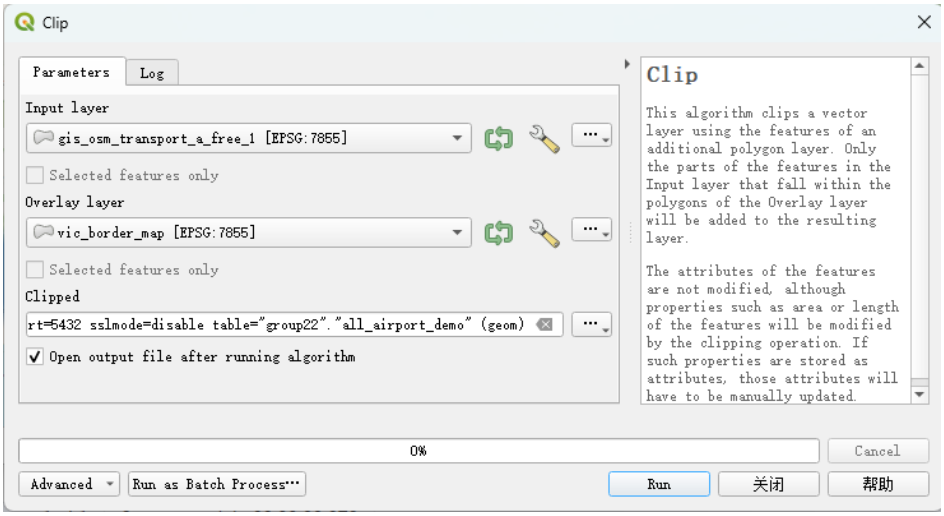
4.1.1 Data Import and Processing

This section provides a detailed introduction to how to import external data from OSM into our PostgreSQL database. Firstly, we use the Database Manager of QGIS to select the group22 schema under the PostgreSQL geom90008 database and click on Import Layer/File Button. Then select the desired layer data as the input, set the Output table, and select the geometry column. Here, you can perform reference system transformation from source data 4326 to project SRID7855, encoded as UTF-8, and establish a spatial index for quick access in the future. The following figure takes importing traffic layers as an example.

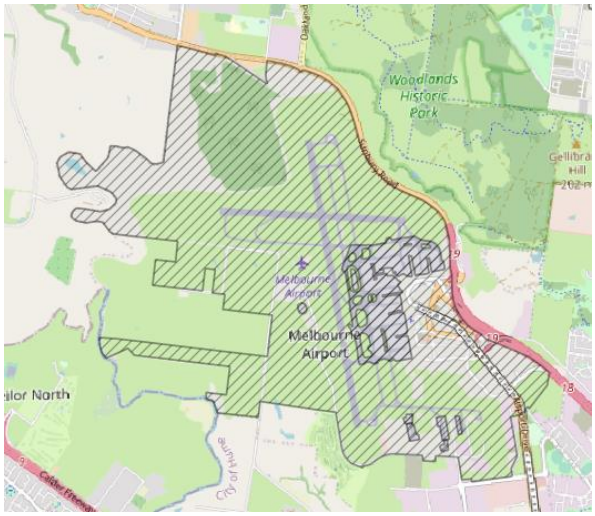


Picture 4.2 Data Import Using QGIS DB Manager

Next, we need to filter based on the required subclass data. In the transport layer, we use airport, airfield, helipad, and apron. We filter these data based on the fclass column, which is $WHERE\ fclass\ IN\ ('airport','airfield','helipad','apron')$.
Furthermore, as the data is Australia wide, we need further processing to obtain data from Victoria. We need to use the Clip tool from QGIS. Select transport data as the input layer, Victoria border data (see authoritative source data in section 4.2) as the overlay layer, and output clipped data as group22. all_ airport table (The table has been merged and deleted with other tables at a later stage) provides spatial data for airports and aprons in Victoria.



Picture 4.3 Data Processing Using QGIS Clip Tool



Picture 4.4 Example: Melbourne Airport

Following the above steps, we filtered and stored all no-fly zone (including airports, aprons, parks, beaches, stadiums, etc.), emergency and hotel spatial data in Victoria from the OSM database. A visualization summary of all external data is provided in Appendix 5.4.

4.2 Authoritative Source Data

The main authoritative source data used in this project include: VicMap, Unimelb Open Spatial Data, and digital boundary files.
The external data provided by VicMap used in this system includes the following tables located in the 'spatial' schema: "victoria_dem_30m", "o_2_victoria_dem_30m", "o_4_victoria_dem_30m", "o_8_victoria_dem_30m", "victoria_roads2023", "victoria_roads2023_vertices_pgr".
VicMap data can be accessed from the following link https://www.land.vic.gov.au/maps-and-spatial/spatial-data/vicmap-catalogueand.vic.gov.au.

Table Names (“spatial” schema)	Description
victoria_dem_30m	The Digital Elevation Model (DEM) data of Victoria is saved in raster format for terrain height. Different prefixes represent different levels of resolution.
o_2_victoria_dem_30m	
o_4_victoria_dem_30m	
o_8_victoria_dem_30m	
victoria_roads2023	Vector data of roads in Victoria, including road names, road geometry data, etc. These data can meet the needs of calculating the shortest path.
victoria_roads2023_vertices_pgr	Vector data of endpoints of roads in Victoria, including endpoint id, endpoint geometry data, etc.
victoria_road_*	Supplementary data on roads in Victoria, including road classification, road direction, etc. Used as reference data.

Table 4.2 Usage of VicMap Data

The data from Unimelb Open Spatial Data used in this system is mainly the spatial data of the University of Melbourne campus, stored in the "unimelb_campus" table under the spatial schema. This table stores the geometry data of various campuses of the University of Melbourne. This data can meet the needs of departing from the school center point to the flight point, and the school center point can be calculated based on the campus geometry data.

The data of Unimelb Open Spatial Data can be obtained from the link https://spatialdata-uom.opendata.arcgis.com/.

Picture 4.5 Spatial data of the Parkville campus in the unimelb_ campus table

The Digital boundary files data contains spatial data on the boundaries of various Australian states, sourced from the Australian Bureau of Statistics and available from https://www.abs.gov.au/statistics/standards/australian-statistical-geography-standard-asgs-edition-3/jul2021-jun2026/access-and-downloads/digital-boundary-files Obtain. We obtain Victorian border data, which can be used to cut all Australian data to obtain interesting data distributed within the state of Victoria.

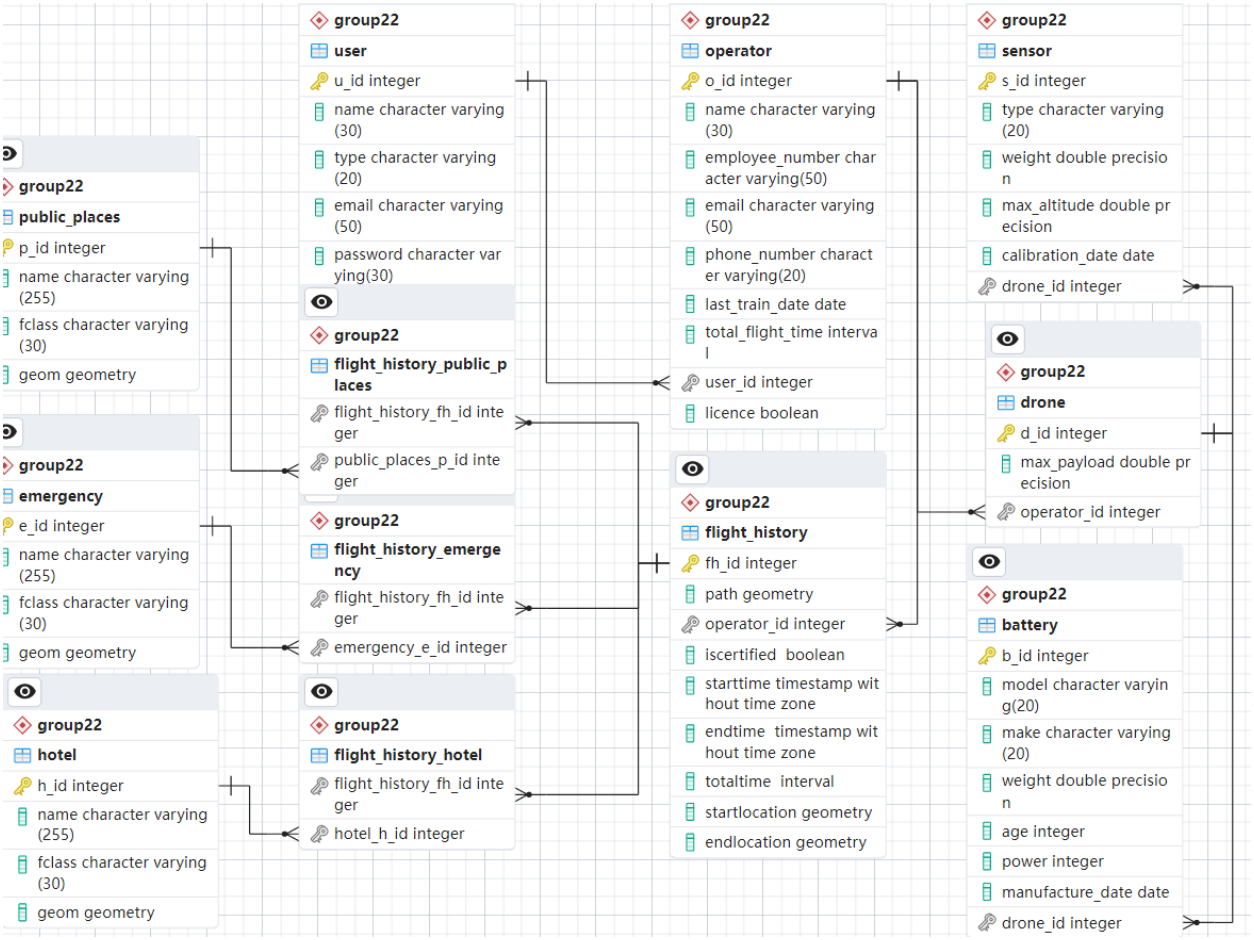
Picture 4.6 Spatial Data at the Victoria State Boundary

4.3 User-Provided Data

Part of the data used in this system is manually provided, including some data in the flight path, drone, battery, sensor, and operator tables.
The Flight path is drawn and generated by the Add Polygon Feature tool of QGIS, stored in the geographic reference system ESPG: 4326, and needs to be transformed into data under the reference system 7855 during calculation.
Part of the data in the Drone, Battery, and Sensor tables is generated by ChatGPT, proofread by the team, and imported into the table.

5. Appendix

5.1 Logical E-R diagram



5.2 Data Dictionary

user (**u_id INTEGER PK**, name VARCHAR (30), type VARCHAR (20), email VARCHAR (30), password VARCHAR (20))

Attribute	Data Type	Description
u_id	INTEGER	User ID (primary key)
name	VARCHAR (30)	User's name
type	VARCHAR (20)	User type (e.g., admin)

email	VARCHAR (50)	User's email address
password	VARCHAR (30)	User's password

operator (**o_id INTEGER PK**, name VARCHAR(30), employee_number VARCHAR(50), email VARCHAR(50), phone_number VARCHAR(20), last_train_date DATE, total_flight_time INTERVAL,***user_id INTEGER FK*** ,licence BOOLEAN)

Attribute	Data Type	Description
o_id	INTEGER	Operator ID (Primary key)
name	VARCHAR (30)	Operator's name
employee_number	VARCHAR (50)	Operator's employee number
email	VARCHAR (50)	Operator's email address
phone_number	VARCHAR (20)	Operator's phone number
last_train_date	DATE	Date the operator last received a training course (format: YYYY-MM-DD, e.g., 2023-04-04)
total_flight_time	INTERVAL	Total accumulated flight time of the operator (unit: hour)
user_id	INTEGER	User ID (Foreign Key referenced User)
licence	BOOLEAN	Determine if the operator

		is qualified to operate a drone (format: Y/N)

drone (**d_id INTEGER PK**, max_payload INTEGER, ***operator_id INTEGER FK***)

Attribute	Data Type	Description
d_id	INTEGER	Drone ID (primary key)
max_payload	DOUBLE PRECISION	Drone's maximum payload (unit: kilogram, e.g., 2.3 kg)
operator_id	INTEGER	Operator ID (Foreign Key referenced Operator)

battery (**b_id INTEGER PK**, model VARCHAR (20), make VARCHAR (20), weight INTEGER, age INTEGER, power INTEGER, manufacture_date DATE, ***drone_id INTEGER FK***)

Attribute	Data Type	Description
ID	INTEGER	Battery ID (primary key)
model	VARCHAR (20)	Battery's model (e.g., B100)
make	VARCHAR (20)	Battery's make (e.g., DJI)
weight	DOUBLE PRECISION	Battery's weight (unit: gram, e.g., 200g)
age	INTEGER	Battery's age (unit: year, e.g., 3 years)
power	INTEGER	Battery's Power (mhA, e.g., 1100mhA)

manufacture_date	DATE	Date of battery production (format: YYYY-MM-DD, e.g., 2023-04-04)
drone_id	INTEGER	Drone ID (Foreign Key referenced Drone)

sensor (**s_id INTEGER PK**, type VARCHAR (20), weight DOUBLE PRECISION, max_altitude DOUBLE PRECISION, calibration_date DATE, ***drone_id INTEGER FK***)

Attribute	Data Type	Description
s_id	INTEGER	Sensor ID (primary key)
type	INTEGER	Sensor Type (e.g., temperature)
weight	DOUBLE PRECISION	Sensor's Weight (unit: gram, e.g., 204.6 g)
max_altitude	DOUBLE PRECISION	Sensor's max working altitude (unit: meter, e.g., 2500m)
calibration_date	DATE	Sensor's last calibration time (format: YYYY-MM-DD, e.g., 2023-04-04)
drone_id	INTEGER	Drone ID (Foreign Key referenced Drone)

flight_history (**fh_id INTEGER PK**, path GEOMETRY, iscertified BOOLEAN, starttime TIMESTAMP, endtime TIMESTAMP, totaltime TIMESTAMP, startlocation GEOMETRY, endlocation GEOMETRY, ***operator_id INTEGER FK***)

Attribute	Data Type	Description
fh_id	INTEGER	Flight history ID (primary

		key)
path	GEOMETRY	Flight's Path (Geometry: LineStringZ 4326)
iscertified	BOOLEAN	Record whether the current operator is qualified to fly (format: Y/N)
starttime	TIMESTAMP	Flight's start time (format: YYYY-MM-DD HH: HI: SS, e.g., 2023-04-04 15:03:02)
endtime	TIMESTAMP	Flight's end time (format: YYYY-MM-DD HH: HI: SS, e.g., 2023-04-04 15:50:10)
totaltime	TIMESTAMP	Flight's total time (unit: minutes., 120min)
startlocation	GEOMETRY	Flight's start location (Geometry: Point)
endlocation	GEOMETRY	Flight's end location (Geometry: Point)
operator_id	INTEGER	Operator ID (Foreign Key referenced Operator)

public_places (**p_id INTEGER PK**, name VARCHAR (255), fclass VARCHAR (30), geom GEOMETRY)

Attribute	Data Type	Description
p_id	INTEGER	Public places ID (primary key)
name	VARCHAR (255)	Public places name
fclass	VARCHAR (30)	Public places fclass
geom	GEOMETRY	Public places Position (Geometry: Point)

emergency (**e_id INTEGER PK**, name VARCHAR (255), fclass VARCHAR (30), geom GEOMETRY)

Attribute	Data Type	Description
e_id	INTEGER	Emergency ID (primary key)
name	VARCHAR (255)	Emergency name
fclass	VARCHAR (30)	Emergency fclass
geom	GEOMETRY	Emergency Position (Geometry: Point)

hotel (**h_id INTEGER PK**, name VARCHAR (255), fclass VARCHAR (30), geom GEOMETRY)

Attribute	Data Type	Description
h_id	INTEGER	Hotel ID (primary key)
name	VARCHAR (255)	Hotel name

fclass	VARCHAR (30)	Hotel fclass
geom	GEOMETRY	Hotel Position (Geometry: Point)

5.3 Demonstration SQL code and result

5.3.1 Away Public facilities SQL

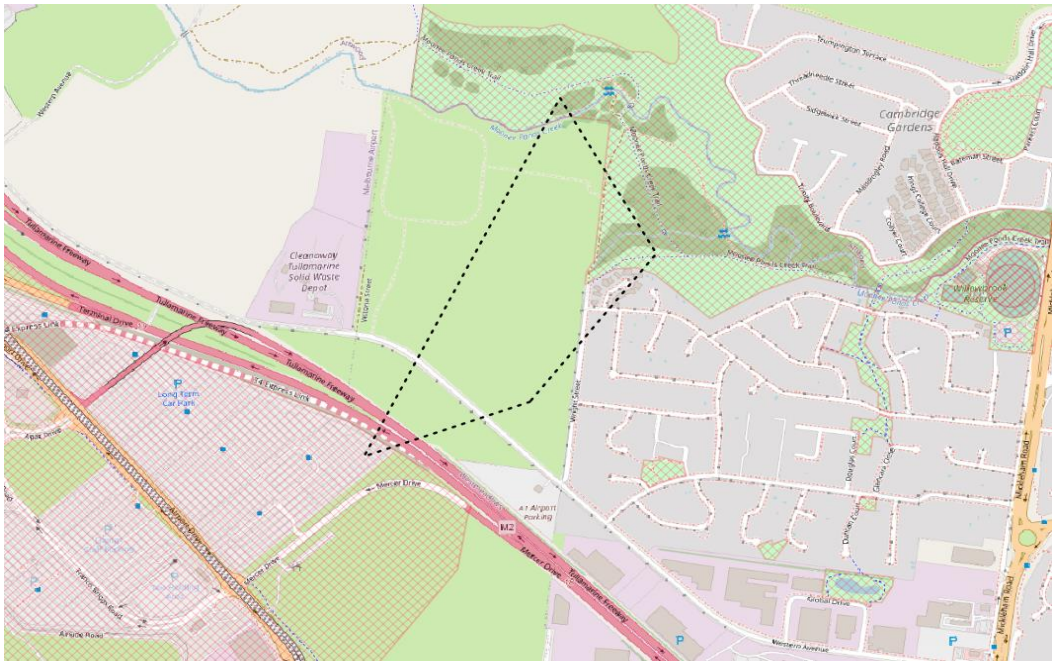
- Code:

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING Z
(144.87103558763067 -37.67814707715162 120,144.8667137848033 -
37.67929871348155 118,144.8720120519403 -37.671311422147866
123,144.87440408410174 -37.6748647958145 130,144.87103558763067 -
37.67814707715162 120)'), 4326) AS geom
)
SELECT *
FROM group22.public_places AS p, flight AS f
WHERE ST_Intersects(p.geom, ST_transform(f.geom, 7855));
```

- Result:
  The flight path passes through Woodland's Historical Park and Melbourne Airport, making the route invalid.

	p_id [PK] integer	name character varying (255)	fclass character varying (30)	geom geometry
1	25351	Woodland's Historical Park	park	0106000020AF1E00000100000001030
2	29757	Melbourne Airport	airport	0106000020AF1E00000100000001030

The black dashed line represents the flight trajectory.



5.3.2 Total weight under max payload SQL

- Code:

```
SELECT d.d_id,
                B.b_id,
                S.s_id,
      d.max_payload,
            -- Two decimal places are retained to ensure accuracy
      ROUND((COALESCE(SUM(B.weight), 0) + COALESCE(SUM(S.weight),
0))::numeric, 2) AS TotalWeight,
            -- Check that the total weight of the batteries and sensors does not
exceed the maximum load of the drone
      CASE
         WHEN d.max_payload >= (COALESCE(SUM(B.weight), 0) +
COALESCE(SUM(S.weight), 0)) THEN 'Within Payload Limit'
         ELSE 'Exceeds Payload Limit'
      END AS PayloadStatus
FROM group22.drone AS d
LEFT JOIN group22.battery AS B ON d.d_id = B.drone_id
LEFT JOIN group22.sensor AS S ON  d.d_id = S.drone_id
WHERE d.max_payload IS NOT NULL --Ignore maximum loads with NULL
values
GROUP BY d.d_id,  d.max_payload,B.b_id,S.s_id
ORDER BY d.d_id;
```

- Result:
  The weight of each drone's corresponding battery and sensor is added up to compare it with the drone's maximum load to obtain its pay load status.

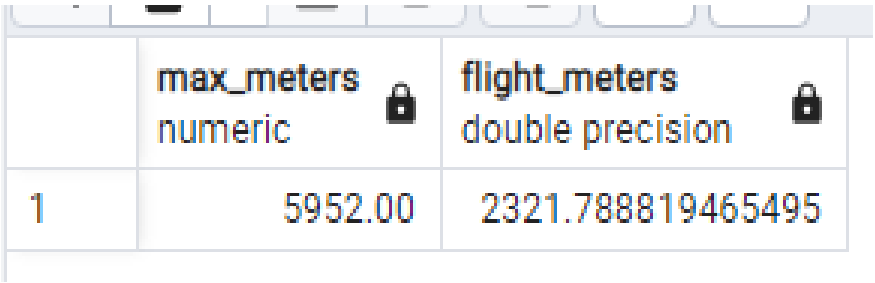
	d_id integer	b_id integer	s_id integer	max_payload double precision	totalweight numeric	payloadstatus text
1	1	9	[null]	2500	196.50	Within Payload Limit
2	2	12	11	3000	495.10	Within Payload Limit
3	3	19	30	2000	364.31	Within Payload Limit
4	4	25	2	3500	401.20	Within Payload Limit
5	5	3	41	4000	357.26	Within Payload Limit
6	6	35	50	2500	560.20	Within Payload Limit
7	7	40	15	3000	439.50	Within Payload Limit
8	8	48	64	1500	579.70	Within Payload Limit
9	9	57	3	2000	402.50	Within Payload Limit

5.3.3 Battery complete route capability detection SQL

- Code:

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING Z
(144.87103558763067 -37.67814707715162 120,144.8667137848033 -
37.67929871348155 118,144.8720120519403 -37.671311422147866
123,144.87440408410174 -37.6748647958145 130,144.87103558763067 -
37.67814707715162 120)'), 4326) AS geom
)
SELECT b.power * 4.96 AS max_meters, ST_Length(ST_Transform(f.geom,
7855)) AS flight_meters
FROM group22.battery AS b, flight AS f
WHERE b_id = 4;
```

- Result:
  According to the battery with ID 4 (model: DJI Phantom 3), calculate the maximum flight distance (in meters) and the flight distance of this flight path (in meters). Based on the comparison, it can be determined that the flight using battery ID 4 can be executed.



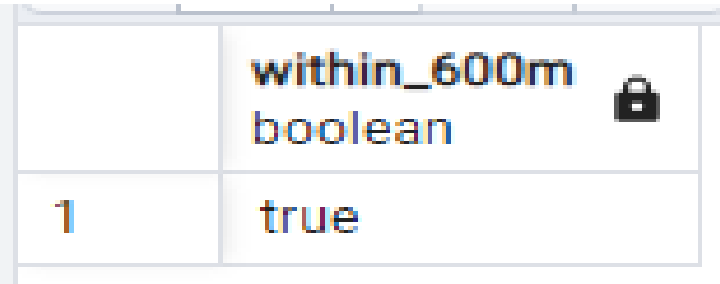
	max_meters numeric	flight_meters double precision
1	5952.00	2321.788819465495

5.3.4 Detection within 600 meters from the starting point SQL

- Code:

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING Z
(144.87103558763067 -37.67814707715162 120,144.8667137848033 -
37.67929871348155 118,144.8720120519403 -37.671311422147866
123,144.87440408410174 -37.6748647958145 130,144.87103558763067 -
37.67814707715162 120)'), 4326) AS geom
), takeoff_point AS (
  SELECT ST_StartPoint(flight.geom) AS geom FROM flight
)
SELECT ST_DWithin(ST_Transform(p.geom, 7855), ST_Transform(f.geom,
7855), 600) AS within_600m
FROM flight AS f, takeoff_point AS p;
```

- Result:
  The flight trajectory for this mission is visible within a visual range of 600 meters from the takeoff point, so the flight plan can be executed.



	within_600m boolean
1	true

5.3.5 Altitude detection SQL

- Code:

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING Z
(144.87103558763067 -37.67814707715162 120,144.8667137848033 -
37.67929871348155 118,144.8720120519403 -37.671311422147866
123,144.87440408410174 -37.6748647958145 130,144.87103558763067 -
37.67814707715162 120)'), 4326) AS geom
), points AS (
  SELECT (ST_DumpPoints(f.geom)).geom AS point
  FROM flight AS f
), max_altitude AS (
        SELECT ST_Z(p.point) - ST_Value(v.rast, ST_Transform(p.point, 7855))
AS max_above_ground_level_meters
        FROM points AS p, spatial.victoria_dem_30m AS v
        WHERE ST_Intersects(ST_Transform(p.point, 7855), v.rast)
        ORDER BY ST_Z(p.point) DESC
        LIMIT 1
), min_altitude AS (
        SELECT ST_Z(p.point) - ST_Value(v.rast, ST_Transform(p.point, 7855))
AS min_above_ground_level_meters
        FROM points AS p, spatial.victoria_dem_30m AS v
        WHERE ST_Intersects(ST_Transform(p.point, 7855), v.rast)
        ORDER BY ST_Z(p.point) ASC
        LIMIT 1
)
SELECT min_altitude.min_above_ground_level_meters,
max_altitude.max_above_ground_level_meters
FROM min_altitude, max_altitude;
```

- Result:

The minimum altitude of this flight path is 6.05 meters, and the maximum altitude is 40.93 meters, relative to the reference ground level. It does not meet the requirement of a minimum altitude of 15 meters, therefore, the execution of this flight path is not permitted.

	min_above_ground_level_meters double precision	max_above_ground_level_meters double precision
1	6.052093505859375	40.92595672607422

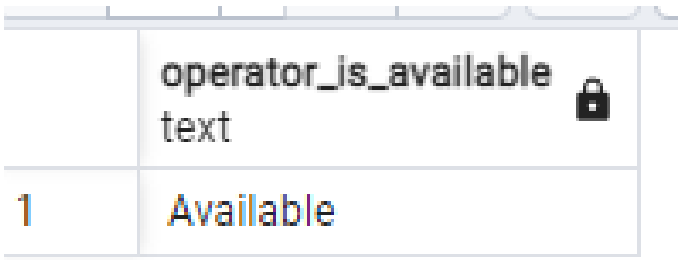
5.3.6 Operator multitasking status detection SQL

- Code:

```
SELECT CASE
    WHEN latest_end_time.endtime < '2023-05-05 12:00:00' THEN 'Available'
    ELSE 'Not Available'
END AS Operator_is_available
FROM (
    SELECT endtime
    FROM group22.flight_history
    WHERE operator_id = 1
          ORDER BY endtime DESC
    LIMIT 1
) AS latest_end_time;
```

- Result:

  The recent end time of the flight for the operator with ID 1 needs to be found in order to compare it with the provided departure time of the new flight plan. If the departure time of the new plan is later than the recent end time of the flight, then the plan is allowed to be executed; otherwise, it is not allowed to be executed. The recent end time of the flight for the operator with ID 1 is "2023-05-04 15:15:02", so it can be assigned to execute the new flight.



	operator_is_available text
1	Available

5.3.7 Unlicenced pilot flight path detection SQL

- Code:

```
SELECT
   f.fh_id,
   f.operator_id,
         o.name,
   o.licence,
   CASE
      -- If the operator has a licence (licence_status = 'Y'), then the flight is always
valid in anywhere
      WHEN o.licence = 'Y' THEN 'Valid Flight anywhere'
      -- If the operator does not have a licence but the flight path is within the
Parkville campus,
                  -- then the flight is valid but only on campus
      WHEN ST_Contains(uc.geom, ST_Transform(f.path, 7899)) THEN 'Valid
Flight only on campus'
      --In all other cases, the flight is not valid
      ELSE 'Invalid Flight'
   END AS FlightStatus
FROM
   group22.flight_history AS f,
   spatial.unimelb_campus AS uc,
   group22.operator AS o
WHERE
         f.operator_id = o.o_id AND
   uc.campus = 'Parkville'
group by
         f.fh_id,
         f.operator_id,
         o.name,
         o.licence,
         FlightStatus;
```

- Result:

  The system will determine what the flight status is based on the licence status of the operator requesting the flight path.

	fh_id integer	operator_id integer	name character varying (30)	licence boolean	flightstatus text
1	1	1	John	true	Valid Flight anywhere
2	2	2	Jane	true	Valid Flight anywhere
3	3	3	Tom	true	Valid Flight anywhere
4	4	4	Kate	true	Valid Flight anywhere
5	5	5	Michael	true	Valid Flight anywhere

5.3.8 Operator data display SQL

- Code:

```
select * from group22.operator;
```

- Result:

	o_id [PK] integer	name character varying (30)	employee_number character varying (50)	email character varying (50)	phone_number character varying (20)	last_train_date date	total_flight_time interval	user_id integer	licence boolean
1	1	John	111	john@example.com	487654321	2022-06-06	52:00:00	1	true
2	2	Jane	112	jane@example.com	412345678	2022-12-23	40:00:00	2	true
3	3	Tom	113	tom@example.com	423456789	2022-05-31	10:00:00	3	true
4	4	Kate	114	kate@example.com	434567890	2022-07-02	30:00:00	4	true

5.3.9 Operators requiring training next month detection SQL

- Code:

```
select * from group22.operator
WHERE licence = true and last_train_date <= NOW() - INTERVAL '1' Year and
total_flight_time< INTERVAL '50' Hour ;
```

- Result:

These are the operator that need training next month.

	o_id [PK] integer	name character varying (30)	employee_number character varying (50)	email character varying (50)	phone_number character varying (20)	last_train_date date	total_flight_time interval	user_id integer	licence boolean
1	3	Tom	113	tom@example.com	423456789	2022-05-31	10:00:00	3	true
2	12	Benjamin	117	benjamin@example.co...	488888888	2022-05-27	12:00:00	17	true
3	14	Nicholas	119	nickolas@example.com	423456789	2022-05-09	35:00:00	19	true
4	27	Jessica	123	jessica@example.com	467891234	2022-05-29	25:00:00	38	true

5.3.10 Battery and sensor maintenance status detection SQL

- Code:

  Battery:

```
select * from group22.battery
WHERE battery.manufacture_date <= NOW() - INTERVAL '2' Year;
```

  Sensor:

```
select * from group22.sensor
WHERE calibration_date  <= NOW() - INTERVAL '1' Year;
```

- Result:

  Battery that manufacture date is more than 2 years ago.

	b_id [PK] integer	model character varying (20)	make character varying (20)	weight double precision	age integer	power integer	manufacture_date date	drone_id integer
1	10	Blade Chroma	Hobbico	104.8	4	1050	2020-10-25	30
2	15	Alpha A32	Sony	125.5	3	1100	2021-04-07	18
3	16	Pulsar 2	Walkera	168.7	4	1300	2020-07-30	25
4	19	Osmo Action	DJI	110	3	1100	2021-02-01	3
5	32	Typhoon Q500 4K	Yuneec	180	3	1200	2021-04-03	21
6	35	S800 EVO	DJI	180	4	1600	2020-05-28	6
7	36	Matrice 600 Pro	DJI	120	3	1100	2021-02-11	[null]
8	37	Phantom 3	DJI	140	4	1200	2020-08-17	28
9	40	DJI Spark	DJI	140	4	1100	2019-11-07	7
10	42	Parrot Bebop 2	Parrot	120	4	1200	2020-06-15	16

  Sensor that calibration date is more than 1 year ago.

	s_id [PK] integer	type character varying (20)	weight double precision	max_altitude double precision	calibration_date date	drone_id integer
1	18	GPS	355.4	5000	2022-05-04	[null]
2	19	motion	273.1	4000	2022-02-08	[null]

5.3.11 Flight history display SQL

- Code:

```
select * from group22.flight_history;
```

- Result:

	fh_id [PK] integer	path geometry	operator_id integer	iscertified boolean	starttime timestamp without time zone	endtime timestamp without time zone	totaltime interval	startlocation geometry	endlocation geometry
1	1	01020000A0E6100000040...	1	true	2023-05-04 15:03:02	2023-05-04 15:15:02	12	[null]	[null]
2	2	01020000A0E6100000050...	2	false	2023-05-05 15:15:02	[null]	[null]	[null]	[null]
3	3	01020000A0E6100000040...	3	true	2023-05-06 09:15:02	2023-05-06 09:32:02	17	[null]	[null]
4	4	01020000A0E6100000060...	4	true	2023-05-07 10:15:02	2023-05-07 10:29:02	14	[null]	[null]
5	5	01020000A0E6100000080...	5	false	2023-05-09 09:15:02	[null]	[null]	[null]	[null]

5.3.12 Nearest 3 hotel and 1 emergency SQL

- Code:

Part 1: Search the three nearest hotels from the flight start point.

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING
Z(144.98241409006857 -37.76301660954355 130,
                    144.98204232805682 -37.76263934715726 130,
                    144.98269314028826 -37.76231143646494 130,
                    144.98324064707032 -37.76202450861073 130,
                    144.9837468670509 -37.761581769551164 130,
                    144.98449055789516 -37.76157366788137 130,
                    144.9849140311435 -37.761655723528946 130,
                    144.9849448891796 -37.76230354014615 130,
                    144.9852236245759 -37.763074389754344 130,
                    144.98526484393165 -37.7635746043922 130,
                    144.98515111864668 -37.764099400292274 130,
                    144.98430414516974 -37.76393528933462 130,
                    144.98342625935595 -37.763443161878804 130,
                    144.982889153999 -37.763369285034 130,
                    144.98241409006857 -37.76301660954355 130)'), 4326)
AS geom
), start_point_on_road AS (
        SELECT v.id
        FROM spatial.victoria_roads2023_vertices_pgr AS v, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <->
ST_Transform(v.the_geom, 7855) ASC
        LIMIT 1
), nearest_3_hotels AS (
        SELECT h.name, h.geom
        FROM group22.hotel AS h, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <-> h.geom
ASC
        LIMIT 3
)
SELECT nearest_3_hotels.name, nearest_3_hotels.geom
FROM nearest_3_hotels;
```

Search the shortest paths to the three nearest hotels from the flight start point.

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING
Z(144.98241409006857 -37.76301660954355 130,
                        144.98204232805682 -37.76263934715726 130,
                        144.98269314028826 -37.76231143646494 130,
                        144.98324064707032 -37.76202450861073 130,
                        144.9837468670509 -37.761581769551164 130,
                        144.98449055789516 -37.76157366788137 130,
                        144.9849140311435 -37.761655723528946 130,
                        144.9849448891796 -37.76230354014615 130,
                        144.9852236245759 -37.763074389754344 130,
                        144.98526484393165 -37.7635746043922 130,
                        144.98515111864668 -37.764099400292274 130,
                        144.98430414516974 -37.76393528933462 130,
                        144.98342625935595 -37.763443161878804 130,
                        144.982889153999 -37.763369285034 130,
                        144.98241409006857 -37.76301660954355 130)'), 4326)
AS geom
), start_point_on_road AS (
        SELECT v.id
        FROM spatial.victoria_roads2023_vertices_pgr AS v, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <->
ST_Transform(v.the_geom, 7855) ASC
        LIMIT 1
), nearest_3_hotels AS (
        SELECT h.name, h.geom
        FROM group22.hotel AS h, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <-> h.geom
ASC
        LIMIT 3
), hotel_points_on_road AS (
        SELECT points.id AS id
        FROM nearest_3_hotels AS h
        CROSS JOIN LATERAL (
                SELECT v.id, st_transform(v.the_geom, 7855) as geom
                FROM spatial.victoria_roads2023_vertices_pgr AS v
                ORDER BY h.geom <-> ST_Transform(v.the_geom, 7855) ASC
```

```
                LIMIT 1)
        AS points
), shortest_path AS (
        SELECT *
        FROM pgr_dijkstra(
                'select "OBJECTID" as id, source, target,
ST_length(st_transform(geom, 7855)) as cost from spatial.victoria_roads2023',

                (SELECT id FROM start_point_on_road),
                ARRAY(SELECT id FROM hotel_points_on_road),
                false
        )
)
SELECT r."OBJECTID" AS id, st_transform(r.geom, 7855) AS geom
FROM spatial.victoria_roads2023 AS r, shortest_path AS p
WHERE p.edge = r."OBJECTID";
```

Part 2: Search the one nearest emergency from the flight start point.

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING
Z(144.98241409006857 -37.76301660954355 130,
                        144.98204232805682 -37.76263934715726 130,
                        144.98269314028826 -37.76231143646494 130,
                        144.98324064707032 -37.76202450861073 130,
                        144.9837468670509 -37.761581769551164 130,
                        144.98449055789516 -37.76157366788137 130,
                        144.9849140311435 -37.761655723528946 130,
                        144.9849448891796 -37.76230354014615 130,
                        144.9852236245759 -37.763074389754344 130,
                        144.98526484393165 -37.7635746043922 130,
                        144.98515111864668 -37.764099400292274 130,
                        144.98430414516974 -37.76393528933462 130,
                        144.98342625935595 -37.763443161878804 130,
                        144.982889153999 -37.763369285034 130,
                        144.98241409006857 -37.76301660954355 130)'), 4326)
AS geom
), start_point_on_road AS (
        SELECT v.id
```

```
        FROM spatial.victoria_roads2023_vertices_pgr AS v, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <->
ST_Transform(v.the_geom, 7855) ASC
        LIMIT 1
), nearest_1_emergency AS (
        SELECT e.name, e.geom
        FROM group22.emergency AS e, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <-> e.geom
ASC
        LIMIT 1
)
SELECT nearest_1_emergency.name, nearest_1_emergency.geom
FROM nearest_1_emergency;
```

Search the shortest paths to the one nearest emergency from the flight start point.

```
WITH flight AS (
  SELECT ST_SetSRID(ST_GeomFromText('LINESTRING
Z(144.98241409006857 -37.76301660954355 130,
                        144.98204232805682 -37.76263934715726 130,
                        144.98269314028826 -37.76231143646494 130,
                        144.98324064707032 -37.76202450861073 130,
                        144.9837468670509 -37.761581769551164 130,
                        144.98449055789516 -37.76157366788137 130,
                        144.9849140311435 -37.761655723528946 130,
                        144.9849448891796 -37.76230354014615 130,
                        144.9852236245759 -37.763074389754344 130,
                        144.98526484393165 -37.7635746043922 130,
                        144.98515111864668 -37.764099400292274 130,
                        144.98430414516974 -37.76393528933462 130,
                        144.98342625935595 -37.763443161878804 130,
                        144.982889153999 -37.763369285034 130,
                        144.98241409006857 -37.76301660954355 130)'), 4326)
AS geom
), start_point_on_road AS (
        SELECT v.id
        FROM spatial.victoria_roads2023_vertices_pgr AS v, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <->
ST_Transform(v.the_geom, 7855) ASC
```

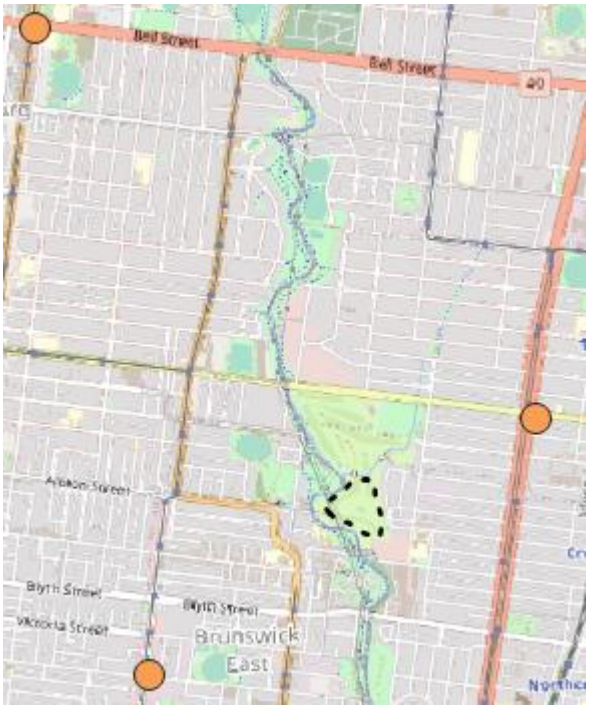
```
        LIMIT 1
), nearest_1_emergency AS (
        SELECT e.name, e.geom
        FROM group22.emergency AS e, flight AS f
        ORDER BY ST_Transform(ST_StartPoint(f.geom), 7855) <-> e.geom
ASC
        LIMIT 1
), emergency_point_on_road AS (
        SELECT points.id AS id
        FROM nearest_1_emergency AS e
        CROSS JOIN LATERAL (
                SELECT v.id, st_transform(v.the_geom, 7855) as geom
                FROM spatial.victoria_roads2023_vertices_pgr AS v
                ORDER BY e.geom <-> ST_Transform(v.the_geom, 7855) ASC

                LIMIT 1)
        AS points
), shortest_path AS (
        SELECT *
        FROM pgr_dijkstra(
                'select "OBJECTID" as id, source, target,
ST_length(st_transform(geom, 7855)) as cost from spatial.victoria_roads2023',

                (SELECT id FROM start_point_on_road),
                ARRAY(SELECT id FROM emergency_point_on_road),
                false
        )
)
SELECT r."OBJECTID" AS id, st_transform(r.geom, 7855) AS geom
FROM spatial.victoria_roads2023 AS r, shortest_path AS p
WHERE p.edge = r."OBJECTID";
```

- Result:
  Part 1: Search the three nearest hotels from the flight start point AND the shortest paths to the three nearest hotels from the flight start point.

  The information about three nearest hotels to the flight start point.

	name character varying (255)	geom geometry
1	St Georges Motor Inn	0101000020AF1E0000A2C2F00D4BBB13413C6E93F9A33256...
2	East Brunswick Hotel	0101000020AF1E00006CDE8709EE9D13411B96DEBA6B3156...
3	Browns Corner Hotel	0101000020AF1E0000501FFAA9459513414BB2DE2D7D3456...

The black dashed line represents the flight route, and the yellow dots indicate the three closest hotels.

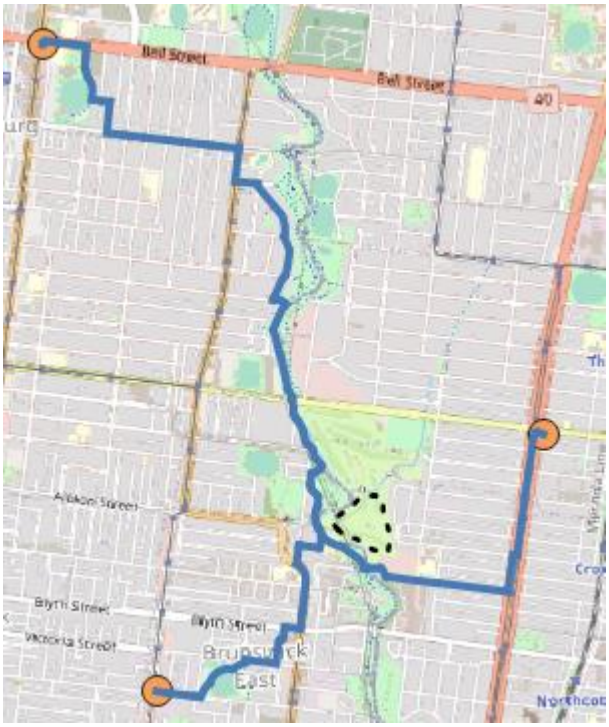


The shortest paths to the three nearest hotels from the starting point of the flight.

	id bigint	geom geometry
1	291524	0105000020AF1E0000010000000102000000040000003C17AB9D78AA
2	396057	0105000020AF1E0000010000000102000000020000000B2BC7D9CBDA8
3	11668	0105000020AF1E0000010000000102000000020000000B2BC7D9CBDA8
4	12536	0105000020AF1E0000010000000102000000250000002913228B90A81
5	54310	0105000020AF1E0000010000000102000000020000000287C56883AA9
6	61614	0105000020AF1E0000010000000102000000020000000082ACA1001A9
7	12960	0105000020AF1E0000010000000102000000020000005C9C695FD8A8
8	1104254	0105000020AF1E0000010000000102000000030000000541F4E6FA8A71
9	60503	0105000020AF1E0000010000000102000000030000000ACCBE1945CA7

Total rows: 71 of 71 | Query complete 00:00:28.137

The blue line segments represent the shortest paths to reach the three nearest hotels from the starting point of the flight.



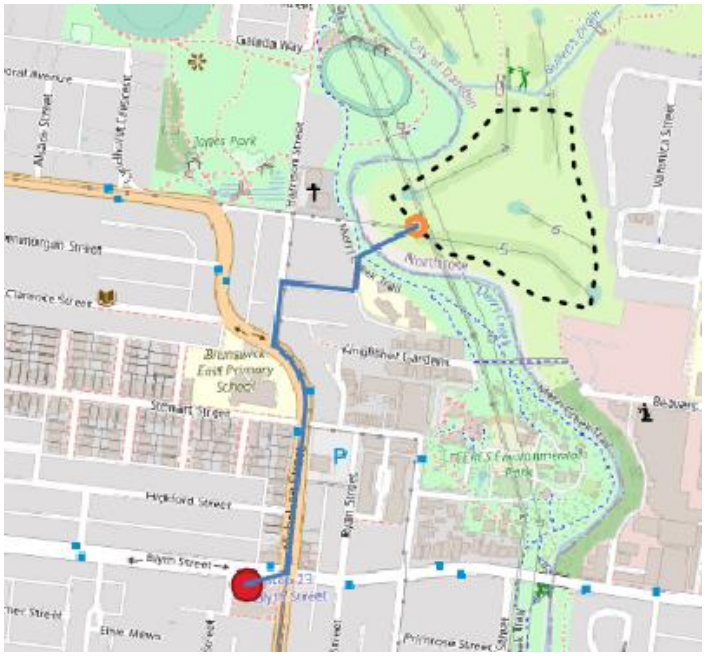
Part 2: Search the one nearest emergency from the flight start point AND the shortest paths to the nearest emergency from the flight start point.

The information about the nearest emergency

	name character varying (255)	geom geometry
1	East Brunswick Medical Centre	0101000020AF1E0000009F4FA4F6A713411AEB965EB33156...

The red dot represents the one nearest emergency from the start point of the flight.

The blue line segments represent the shortest path to reach the one nearest emergency from the starting point of the flight.



5.3.13 Shortest path to start point SQL

- Code:

```
WITH centralpoint AS (
 SELECT ST_PointOnSurface(ST_Union(st_transform(geom,7855))) AS geom
FROM spatial.unimelb_campus
 WHERE campus = 'Parkville'
), central_nearest_point_on_road as (
 SELECT v.id as id, st_transform(v.the_geom,7855) as geom
```

```
 FROM centralpoint AS c, spatial.victoria_roads2023_vertices_pgr AS v
 ORDER BY ST_Distance(c.geom, st_transform(v.the_geom,7855))
 LIMIT 1
), flight_start_point as (
 select st_startpoint(geom) as geom from group22.demo_flight_path
), nearest_point_on_road as (
 select v.id as id, st_transform(v.the_geom, 7855) as geom
 from flight_start_point as p, spatial.victoria_roads2023_vertices_pgr as v
 order by st_distance(p.geom, st_transform(v.the_geom, 7855))
 limit 1
), paths as (
 select * from pgr_dijkstra(
 'select "OBJECTID" as id, source, target, st_length(st_transform(geom,7855)) as
cost from spatial.victoria_roads2023',
 (SELECT id FROM central_nearest_point_on_road),
 (SELECT id FROM nearest_point_on_road),
 false
 )
)
select st_transform(r.geom, 7855) as geom
from spatial.victoria_roads2023 as r, paths as p
WHERE p.edge = r."OBJECTID";
```

- Result:
  This is the shortest path from central to the start point.

	geom geometry
1	0105000020AF1E0000010000000102000000020000000CA2E7C8A4790134162D6DF918B2E56416A7D1BB3FF8E1341B02BC3C18D2E5641
2	0105000020AF1E0000010000000102000000070000006A7D1BB3FF8E1341B02BC3C18D2E56416694D29EDD8E134118EF1CF68D2E5641605DBBAAC88E
3	0105000020AF1E0000010000000102000000020000000F3B30602D38C13410F8143D88F2E5641AF585B21D08B134170C1BFAA912E5641
4	0105000020AF1E0000010000000102000000020000000AF585B21D08B134170C1BFAA912E5641142D40BA998B13410859B0B7922E5641
5	0105000020AF1E0000010000000102000000020000002564B97BBF8B1341B2EFEB9EB12E5641142D40BA998B13410859B0B7922E5641
6	0105000020AF1E0000010000000102000000020000002D1F80DFD78B1341EA1796D5C32E56412564B97BBF8B1341B2EFEB9EB12E5641
7	0105000020AF1E0000010000000102000000020000005E867413E28B1341F01F4D5CD12E56412D1F80DFD78B1341EA1796D5C32E5641
8	0105000020AF1E0000010000000102000000020000020431F15A78B1341C282D16ED12E56415E867413E28B1341F01F4D5CD12E5641
9	0105000020AF1E0000010000000102000000020000B55B4952708B134121A177B1D12E564120431F15A78B1341C282D16ED12E5641
10	0105000020AF1E0000010000000102000000020000DECEE528A08A134132F3EBAED22E5641B55B4952708B134121A177B1D12E5641
11	0105000020AF1E0000010000000102000000020000001437AD22EC891341263F1C8AD32E5641DECEE528A08A134132F3EBAED22E5641
12	0105000020AF1E0000010000000102000000020000EC22648D0689134140DEA4A1D42E56411437AD22EC891341263F1C8AD32E5641
13	0105000020AF1E0000010000000102000000020000006[illegible]


Albion Street
Anstey
Blyth Street
Brunswick
Victoria Street
Brunswick East
Glenlyon Road
Jewell
Princes Hill
Fitzroy North
Carlton North
Melbourne General Cemetery
Rathdowne Street
City of Yarra
83
Elgin Street
Carlton
Fitzroy

5.3.14 Device info display

- Code:

```
select * from group22.sensor;
Select * from group22.drone;
select * from group22.battery;
```

- Result:

This is all the sensor data.

	s_id [PK] integer	type character varying (20)	weight double precision	max_altitude double precision	calibration_date date	drone_id integer
1	1	temperature	204.6	3000	2023-01-13	[null]
2	2	humidity	231.2	2000	2023-04-02	4
3	3	pressure	212.5	4000	2023-02-28	9
4	4	wind speed	304.8	3000	2023-01-23	[null]
5	5	light intensity	244.3	2000	2023-03-17	13
6	6	CO2	218.1	4000	2023-02-11	[null]
7	7	sound	329.6	3000	2023-01-09	21
8	8	GPS	278.5	5000	2023-05-01	[null]
9	9	motion	249.4	2000	2023-03-22	30
10	10	vibration	276.9	3000	2023-01-05	[null]
11	11	temperature	321.7	4000	2023-02-14	2
12	12	humidity	398.6	3000	2023-01-30	[null]
13	13	pressure	374.2	5000	2023-05-08	[null]

This is all the drone data.

	d_id [PK] integer	max_payload double precision	operator_id integer
1	1	2500	3
2	2	3000	1
3	3	2000	9
4	4	3500	14
5	5	4000	18
6	6	2500	10
7	7	3000	22
8	8	1500	6
9	9	2000	11
10	10	3000	12
11	11	3500	20
12	12	4000	7

This is all the battery data.

	b_id [PK] integer	model character varying (20)	make character varying (20)	weight double precision	age integer	power integer	manufacture_date date	drone_id integer
1	1	B100	DJI	140.5	3	1100	2022-01-01	23
2	2	Tattu Plus	Tattu	183.2	4	1300	2022-06-10	[null]
3	3	High-Performance	Yuneec	102.9	3	1050	2023-03-21	5
4	4	Phantom 3	DJI	115.8	4	1200	2022-08-15	[null]
5	5	Mavic 2	DJI	170.3	4	1250	2021-09-03	17
6	6	PowerCell	Tattu	150.7	3	1150	2022-02-28	[null]
7	7	Zenmuse	Inov8	185.9	4	1400	2021-11-11	[null]
8	8	Spark	Flywoo	132.1	3	1100	2022-05-01	[null]
9	9	Inspire 1	DJI	196.5	3	1300	2022-03-15	1
10	10	Blade Chroma	Hobbico	104.8	4	1050	2020-10-25	30
11	11	Evolve 2	DJI	120.6	3	1150	2022-02-01	[null]
12	12	Gremsy S1	SkyHd	173.4	4	1200	2021-08-05	2
13	13	Cinestar 8	FreeFly	189.8	3	1350	2023-01-15	32

5.3.15 Flight plan daylight detection SQL

- Code:

```
SELECT
  CASE
    -- Assuming that the drone can fly during daylight hours of 6:00 to 17:00
    WHEN EXTRACT(HOUR FROM NOW()) BETWEEN 6 AND 17 THEN 'Can Fly'
    ELSE 'Cannot Fly'
  END AS CanFlyStatus;
```

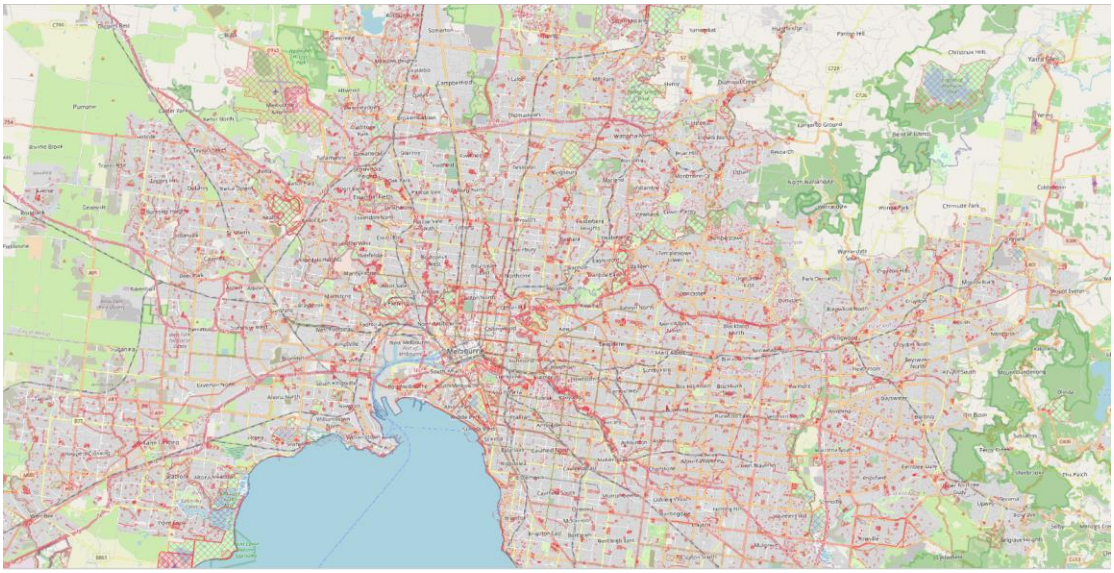
- Result:

We set the daytime to be between 6:00 and 17:00, both in winter and summer, and users can fly during this time.

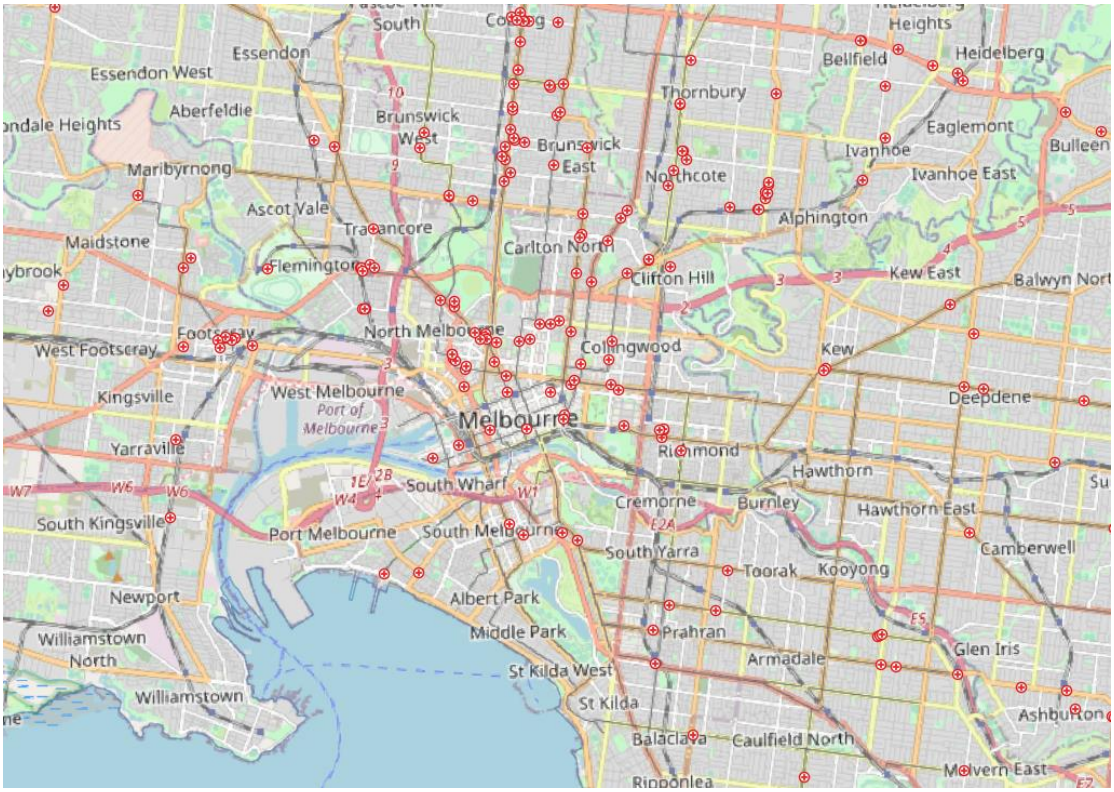
	canflystatus text
1	Can Fly

5.4 External Data Snapshot

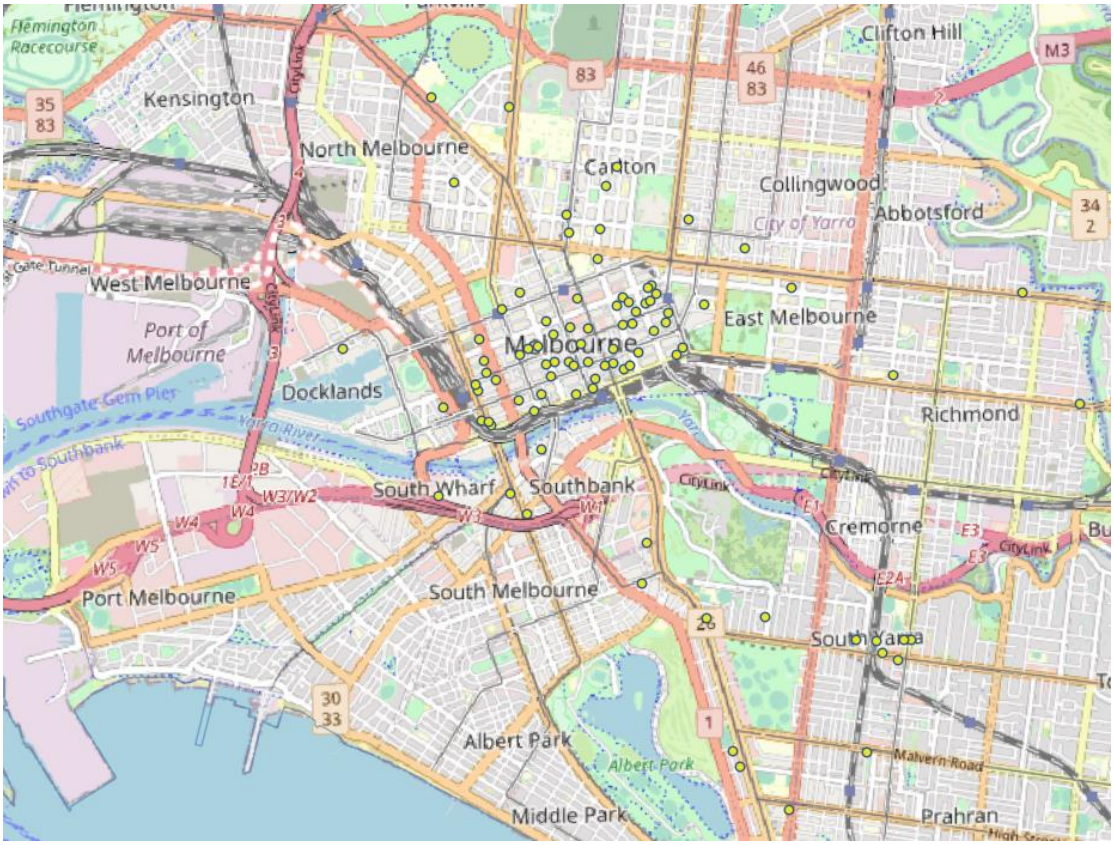
Overview of the drone no-fly zone in Victoria (Melbourne area intercepted). The no-fly zone includes the airport, apron, park, beach, stadium, etc. The red shaded area in the figure is the no-fly zone.



Overview of emergency in Victoria (Melbourne area intercepted). Emergency includes hospitals, emergency centers, clinics, etc. The cross pattern in the figure represents the emergency location.



Overview of accommodation in Victoria (Melbourne area intercepted). Accommodation includes hotels and motels. The yellow dots in the picture indicate the accommodation location.



6. Individual Reflection

6.1. Jiaming Xian

In the implementation phase of the project, I am mainly responsible for: 1) searching, importing, and processing external data, as well as writing corresponding modules in the report; 2) Writing SQL related to functions and implementing QGIS visualization; 2) Sort out and write the data dictionary and specifications, and write the corresponding modules in the report.

I actively participate in group discussions and collaborations, and believe that our group collaboration has been very successful. Firstly, our group has held many meetings, with 1-2 meetings per week, each lasting at least 3 hours. During the meeting, each group member can contribute to the discussion. We have gathered collective wisdom to complete this project, from proposing a proposal to designing a database and then implementing a database, where everyone has contributed their talents and achieved effective collaboration.

I also evaluated my strengths through the collaborative process. I gained a lot from group collaboration. In collaboration, I play the role of knowledge provider, coordinator, and reviewer, which is related to my own strengths. Firstly, I am very proficient in the knowledge points and skills of photography and practical, and the additional self-learning has also enabled me to master a lot of spatial data/PostgreSQL/QGIS skill reserves, allowing me to provide rich knowledge in group collaboration. This comes from my self-learning ability. Secondly, my calm personality allows me to play the role of coordinator in group collaboration. When the group falls into debate or disagreement, I am able to accommodate multiple opinions and coordinate discussions to reach further consensus. Furthermore, my careful and diligent approach can help the team avoid making human errors, including unclear understanding of project requirements, confusion of knowledge points, and written input errors. The above is my reflection on my own strengths in the project.

My weaknesses are also reflected through collaboration and actively remedied through my personal efforts. Firstly, I lack practical experience in databases and knowledge related to spatial data at the project site. My remedy is to closely follow the lecture materials, practice based on practical materials, and supplement my missing knowledge and skills by asking questions on tutorial and self-learning through search engines. Secondly, in collaboration, I am not good at raising questions or refuting opinions due to my gentle personality. My remedy is to boldly express my own views and bravely ask and refute certain opinions that make me suspicious. This is because I have realized through collaboration that only when everyone's viewpoints collide can we reach a better consensus and produce products that better align with everyone's ideas in subsequent project implementations. On the contrary, if everyone does not provide suggestions, the project output will be unimaginably poor because there is no collective wisdom to contribute to it.

In summary, in GEOM9008 Spatial Data Management course, I learned knowledge and skills related to spatial data, including but not limited to vector/raster data, routing algorithms, postgreSQL databases, QGIS software, spatial correlation functions, etc. And through group collaboration projects, I have further improved my ability to collaborate with others, honed my coordination skills, and had the courage to express my opinions. In the future, I will be able to better play my role in collaboration, contribute to the team, and continue to maintain self-directed learning, laying a solid foundation for future industry needs.

6.2. Sijia Pei

This group assignment is about the final document displayed in the database. Because we completed most of the database creation work in advance, in this assignment, team members only need to complete the document part of the work.

For me personally, there are some contributions to the project and some shortcomings that I have made.

Firstly, for the team, I participated in some tasks and my main contributions were:

● I redrawn the er diagram of our latest database

● I updated the Data Dictionary of the er graph

● I have conducted a detailed analysis of the newly updated database structure and explained why our database was created in this way, as well as why the relationships between each table are connected in this way

● I have found two methods to improve database performance, index and trigger

● I created triggers and indexes in the database and tested the improvement in database performance using them

During this process, I also discovered some obvious areas where I am not good at:

● I am still not very proficient in SQL and it took me a lot of time to write a few SQL statements

Although not doing very well in this aspect, with the help of team members, we solved this problem and ultimately completed the task of this stage together.

For future project development, if there is still an opportunity to continue developing the project, then I believe we can find more detailed data to meet the functions beyond our scope. We have currently implemented most of the functions, and a few are unable to achieve them due to the lack of data sources that can help us achieve those functions. Our group assignment this time still did not improve much efficiency, and the disorganized meeting process delayed everyone's time. Unsmooth group work is also a very important work experience, but after these small difficulties, I have also learned a lot about how to get along with other development teams in work.

This project not only improved my ability in project management and teamwork, but also strengthened my knowledge in databases. In my previous assignments, I have become proficient in how to write data documents, which has been very helpful to me and has been a valuable experience for my future studies and work.

I believe that both the subsequent development of this project and my future development in the industry will benefit from it. If there is still an opportunity, I look forward to working with my team members to optimize our database system again in additional time.

6.3. XINGTING Zong

During the development of that project, I worked in a variety of different types of positions, which I will elaborate on as my personal reflections in the following points:

Firstly, I am the main initiator and organizer of the team's development project meetings, in which I plan and manage the dates and content of the meetings based on my previous project management knowledge and experience, to ensure that each member is on time and active in the development discussions. As our team was in an online mode, it was a huge challenge to schedule and organize the meetings as it was important to balance everyone's workload with the time available for meetings, especially the exchange of information and communication between each other. Fortunately, our project was eventually completed successfully, but I think my project management and organizational experience could still be improved. In future team development projects, I need to be aware of the need to optimize the content and frequency of project meetings and to reduce unnecessary meetings to save time. I also need to be more proactive in communicating with team members to ensure that there is no confusion among project members and to avoid project development problems.
Secondly, as part of the development team, I was involved in all aspects of the project, including project reporting, project database development, project data source search and research, project video editing, and project functionality development. Throughout the development cycle, I was not only the main developer, but I also supported the others in their development work, such as collating data, preparing images for data visualization, and other 'minor' tasks. The main purpose of this support work was to bridge the development between team members to ensure that the final presentation was a complete package, such as a report. I think this kind of support work is necessary for the team development process, as each person on the team will have a different idea. In future projects, the way in which the details and interfaces are handled will be crucial to the success of the project, and this is a challenge to my personal skills and experience, which I am happy to face and do 'behind the scenes'. But in the process, it is important to maintain careful communication with the other members to ensure that their intentions are not misinterpreted, otherwise, there will be errors in detail and interface.

My personal strengths are my diversity and motivation. I can work on any type of development and deliver quality and quantity, such as database development, report writing, video editing, data research, etc. I am also good at using my motivation to drive projects forward and motivate the team to ensure that projects are completed ahead of schedule and are tested. My personal weakness is that I am not very subjective and am easily influenced by the ideas of other team members. This is my biggest challenge when developing projects and it leads me to sometimes let myself get caught up in other people's ideas and not be able to make an accurate judgment and have to test each project to get a final decision.
Through my personal strengths and weaknesses, as well as my role and contribution in this project development, I believe that my performance in this project development process was satisfactory and the development process was basically smooth, although there were some minor problems, none of them affected the project development process. The minor issues that arose were converted into lessons learned for me so that I will not make the same minor mistakes in the future.

6.4. YANXI Ke

In this group of projects, I am responsible for project planning, function analysis, and function realization. I was actively involved in all phases of the project. Specifically, it has completed many functions such as drone route detection (relevant map display), drone, sensor, battery management, etc. During the whole process, I discovered some of my strengths and weaknesses, and constantly improved myself through cooperation with team members.

First of all, I found that I have strong communication and coordination skills. Good at listening to the opinions and suggestions of others, able to summarize various opinions and viewpoints, so that the team can better reach a consensus. I am also willing to help others and try my best to resolve the contradictions within the team, so that the team always maintains a good cooperative atmosphere. At the same time, I can also communicate the latest progress of the project to the team members in time to ensure that everyone understands the progress of the project.

Secondly, I also have certain abilities in self-reflection and learning. I am open to feedback and guidance from others and am able to apply this information to improve myself. When I first started implementing the project functions, I may not have paid enough attention to some details of the UAV CASA rules, but with the help of team members and self-study, I gradually became more careful and diligent. I know that personal and professional development requires constant learning and growth, so I am constantly improving my skills and knowledge.

However, I also realize that I still have room for improvement in some aspects. For example, in the early stage of the project, I didn't have a deep understanding of database-related Network Analysis knowledge, so I needed to ask and learn from team members. In future projects, I hope to be able to supplement and expand my knowledge base more autonomously, so as to better deal with various challenges. But with the study of this project, I think I made up for this deficiency very well.

Overall, this group project has been very rewarding for me. By demonstrating my communication and coordination skills and a self-reflective learning attitude, I am able to be the cohesive force of the team and actively resolve conflicts and problems. In my future personal and professional development, I will continue to focus on improving communication and collaboration skills, deepening professional knowledge, maintaining a growth mindset, and developing project management capabilities. I believe that through continuous efforts and improvement in these key areas, I will be able to overcome various challenges and achieve greater success in team projects and individual work.