BT5110 Project StarAIS

Data Warehouse for AIS Messages

Group 3

Ong Jing Wen James (A0217260N)

Ang Guang Jun Nicholas (A0217246H)

Anshumaan Phukan (A0280562E)

Ng Jing Yi (A0159757L)

National University of Singapore

Master of Science in Business Analytics Programme

10 November 23

1. Introduction

Singapore is a pivotal maritime hub and is one of the world's busiest ports. She is situated at the Southeast Asian Maritime corridor and facilitates a nexus for global trade routes. Automatic Identification System (AIS) messages promote maritime navigation and safety as insights into vessel movements and behavior in the maritime industry. This report examines a dataset from *Data@Liánchéng*, comprising a five-year collection of AIS messages from the western Singapore Strait. The data can be extracted for each month. In this report, we present our data extraction and transformation methodology database schema and perform queries through a user interface (UI). The results from the queries show that our database and UI can be used to elucidate patterns and trends in maritime traffic, providing insights to industry stakeholders. Our application can also be developed for future maritime planning and operations.

2. Methodology

2.1 Extracting AIS Message Data from Data@Liánchéng (Q1). This section describes the approach employed in data extraction and analyses. Initially, we aimed to identify a dataset timeframe with minimal atypical maritime traffic behaviours, such as the impact of seasonal deviations and the pandemic period between 2020 and 2021. For instance, shipping traffic may increase in Q4 due to Black Friday, 10.10 and Christmas. In contrast, shipping traffic may decrease in Q1 during the Chinese New Year period (January to February) due to factory shutdowns in Asia. The COVID-19 pandemic may have caused unstable maritime patterns, a decrease in trade at the onset of the pandemic, and an increase in trade towards the end of 2021. However, we could extend the project to include these periods for supply chain and logistics-related capabilities, such as real-time supply chain visibility.

We also factored in the construction of the star schema when choosing which dataset to extract. Conducting the exploratory data analysis (EDA), which showed many missing values to be recovered from external data sources, such as through web scraping. Therefore, we decided to extract the August 2023. August 2023 data will fulfill the abovementioned considerations and the vessels have a higher probability of still being in service, increasing the likelihood of success when web scraping for additional information.

After establishing the temporal parameters, we concentrated on AIS message types that will provide meaningful data for our analyses. Specifically, we chose type 1 and type 5 messages to populate our data warehouse. Type 1 messages provide position report messages, comprising most of our dataset with approximately 11 million unique messages detailing the vessel and time of message transmission and other measures such as turn, accuracy, speed, longitude, and latitude. Type 5 messages are static and voyage-related data, providing comprehensive vessel characteristics, such as the ship's unique nine-digit maritime mobile service identity (MMSI) number, unique seven-digit international maritime organization number (IMO), vessel name, vessel type, and its dimensions. Type 5 messages shed light on the ships in our dataset.

2.2 Data Staging Area: Transforming and Loading Data (Q2 & 3)

This section describes the creation of our data warehouse, which is the foundational step for this project. Firstly, we designed our star schema to segregate our dataset into fact and dimension tables systematically.

2.2.1 Fact Table

Our fact table is the primary table of our data warehouse, which captures each navigational message to the finest level of detail. Type 1 messages serve as the primary data source for this table. Every row within the fact table represents an individual message dispatched by a vessel, providing quantitative measures that can be summed or averaged for post-processing and analytics. The table includes keys referencing our dimension tables to ensure referential integrity and enable complex queries through joins.

First, we identified several measures within our fact table: `accuracy`, `course`, `heading`, `speed`, and `turn`. Among 11,362,710 type 1 messages, we deemed entries with missing `speed`, `course`, and `heading` incomplete and subsequently omitted to streamline query operations. As a result, our distilled fact table comprises 2,332,115 rows. Table 1 shows the message fact table.

Message Fact Table
message_id
location_id
ship_id
datetime_id
status_id
accuracy
course
heading
speed
turn

Table 1: Message Fact Table

2.2.2 Surrogate keys

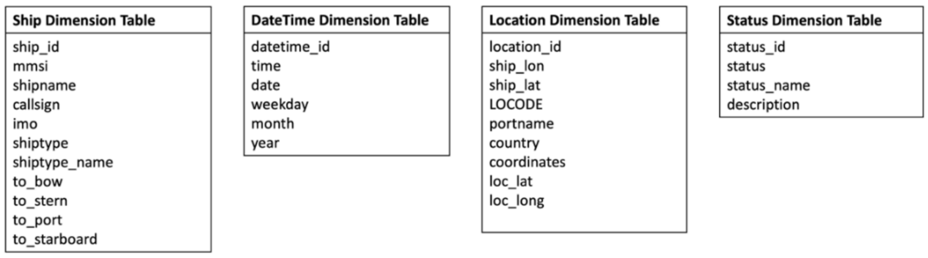
Second, we generated surrogate keys for the message fact table. In particular, `message_id` is a unique identifier for each message to maintain order and facilitate data retrieval. This key was serially generated from index 1 for every message in the message fact table. `location_id`, `ship_id`, `datetime_id`, and `status_id` are also surrogate keys that are unique to every row in the location, ship, datetime, and status dimension table, respectively. This was accomplished by dividing the type 1 messages into distinct data frames, each corresponding to the different dimension tables of our schema. Specifically,

- Location information (longitude and latitude) was allocated to the location dimension table
- Status was segregated into the status dimension table
- Vessel information was categorised into the ship dimension table

After allocating data to the respective dimension tables and cleansing to ensure record uniqueness, we similarly numbered each row serially starting from 1 to generate surrogate keys. Although the data warehouse in our project only encompassed August 2023 data, we designed the infrastructure for scalability through the surrogate key system. These keys are integral to allowing for future expansion on the warehouse without disrupting the existing data structure. They are also pivotal in buffering our data warehouse against operational changes. Detailed descriptions of all keys and measures within the fact table are documented in Appendix A1 to support the transparency and usability of our data warehouse.

2.3 Creating and populating the dimension table (Q3)

This section details the creation and population of our dimension tables. Figure 1 shows the four-dimension tables described in Table 1, comprising ship, location, status, and datetime. In brief, the data is derived from the AIS data and from scraping VesselFinder, and Figure 2 shows the dataflow architecture after pre-processing.



Ship Dimension Table
ship_id
mmsi
shipname
callsign
imo
shiptype
shiptype_name
to_bow
to_stern
to_port
to_starboard

DateTime Dimension Table
datetime_id
time
date
weekday
month
year

Location Dimension Table
location_id
ship_lon
ship_lat
LOCODE
portname
country
coordinates
loc_lat
loc_long

Status Dimension Table
status_id
status
status_name
description

Fig 1: Dimension Tables

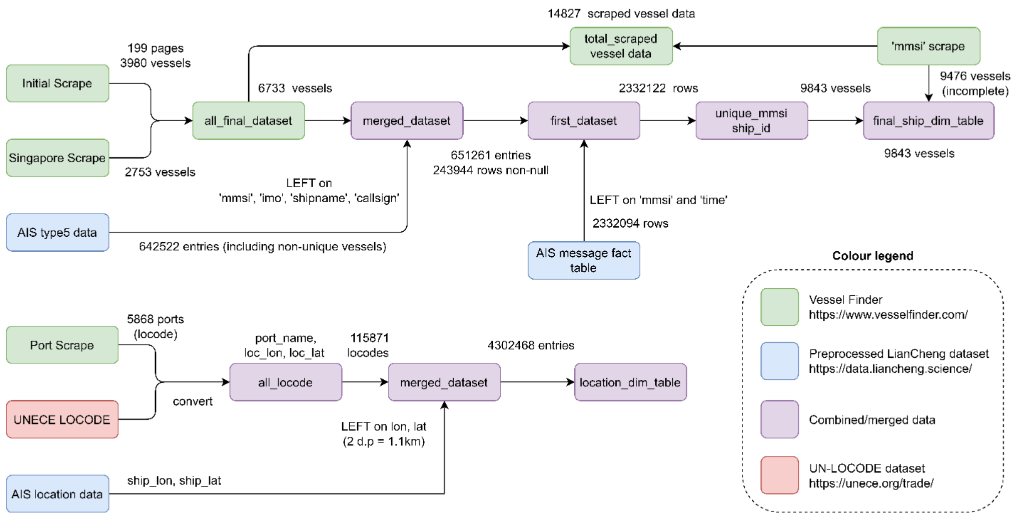


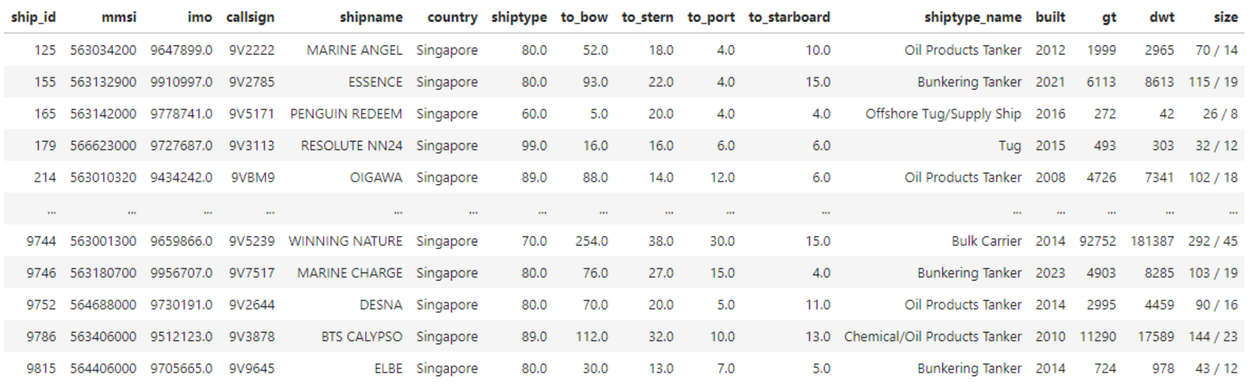
Fig 2: Dataflow Architecture

2.3.1 Location Dimension Table

The location dimension table is constructed from longitudinal and latitudinal data derived from AIS type 1 messages. We allocated a unique `location_id` for each longitudinal and latitudinal pairing to facilitate the retrieval of vessel-specific geographic positions. This data is pivotal for tracking vessel positions, optimizing navigational routes, and analysing spatial trends in maritime activities. The initial dataset was extracted from VesselFinder, which provided information on the port attribute, giving a total of 5868 locations, port names, countries, and location codes (LOCODE). Subsequently, we matched the LOCODEs in the open-source UN/LOCODE dataset to provide a location code of vessels outside the ports. For example, SGSIN is the LOCODE for the Port of Singapore recorded in VesselFinder. However, vessels could also dock around Sebarok Island (SGSEB) or Pasir Panjang Wharves (SGKEP). From the UN/LOCODE dataset, the North-South-East-West (NSEW) coordinates were obtained and converted into Decimal Degrees (DD) format for longitude and latitude, yielding 115,871 unique LOCODEs. Integrating this with AIS location data from type 1 messages and approximating the DD to two decimal points, we achieved a precision of roughly 1.1km to the closest LOCODE. Thus, the enabled correlation of port locations with ship positions allows us to conduct advanced queries such as determining a vessel's port proximity. Figure 1 shows the attributes of the resulting location dimension table, and the documentation is presented in Appendix A2.

2.3.2 Ship Dimension Table

The attributes in the ship dimension table contain data on the physical dimensions and other identifying information about the vessels. Our initial data extraction covered 3980 vessels. However, merging with the AIS type 5 data yielded insufficient information to gain actionable insights. Further EDA of the locations dimensions table revealed a significant concentration of vessels within Singapore's territorial waters. As such, we conducted another data collection effort for Singapore-registered vehicles, culminating in our dataset of 6733 vessels. After joining the AIS message fact table on MMSI and temporal markers, 9843 unique ship identifiers (`ship_ids`) were created. However, a mere 259 vessels were matched from our dataset. Assuming the persistence of MMSI at the AIS data extraction juncture, we conducted an additional scrape for 9476 vessels based on their MMSI. The cumulative data acquisition effort resulted in a dataset of 14827 vessel records, forming the final ship dimension table. The full explanation of all the keys and measures in our ship dimension table can be found in Appendix A3.



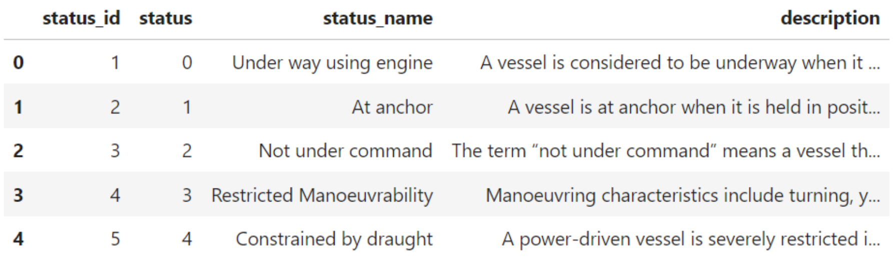
ship_id	mmsi	imo	callsign	shipname	country	shiptype	to_bow	to_stern	to_port	to_starboard	shiptype_name	built	gt	dwt	size
125	563034200	9647899.0	9V2222	MARINE ANGEL	Singapore	80.0	52.0	18.0	4.0	10.0	Oil Products Tanker	2012	1999	2965	70 / 14
155	563132900	9910997.0	9V2785	ESSENCE	Singapore	80.0	93.0	22.0	4.0	15.0	Bunkering Tanker	2021	6113	8613	115 / 19
165	563142000	9778741.0	9V5171	PENGUIN REDEEM	Singapore	60.0	5.0	20.0	4.0	4.0	Offshore Tug/Supply Ship	2016	272	42	26 / 8
179	566623000	9727687.0	9V3113	RESOLUTE NN24	Singapore	99.0	16.0	16.0	6.0	6.0	Tug	2015	493	303	32 / 12
214	563010320	9434242.0	9VBM9	OIGAWA	Singapore	89.0	88.0	14.0	12.0	6.0	Oil Products Tanker	2008	4726	7341	102 / 18
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9744	563001300	9659866.0	9V5239	WINNING NATURE	Singapore	70.0	254.0	38.0	30.0	15.0	Bulk Carrier	2014	92752	181387	292 / 45
9746	563180700	9956707.0	9V7517	MARINE CHARGE	Singapore	80.0	76.0	27.0	15.0	4.0	Bunkering Tanker	2023	4903	8285	103 / 19
9752	564688000	9730191.0	9V2644	DESNA	Singapore	80.0	70.0	20.0	5.0	11.0	Oil Products Tanker	2014	2995	4459	90 / 16
9786	563406000	9512123.0	9V3878	BTS CALYPSO	Singapore	89.0	112.0	32.0	10.0	13.0	Chemical/Oil Products Tanker	2010	11290	17589	144 / 23
9815	564406000	9705665.0	9V9645	ELBE	Singapore	80.0	30.0	13.0	7.0	5.0	Bunkering Tanker	2014	724	978	43 / 12

Figure 3: Sample of ship dimension table

2.3.3 Status Dimension Table

The status dimension table in our data warehouse is constructed from the navigational status information derived from type 1 messages. The AIS navigational status is indicated by a numerical code ranging from 0 to 15, with each code representing a distinct operational state of the vessel. We correlated each numerical status with its respective name and description to render these codes to our users. The status information is essential for operational monitoring, safety compliance, and logistical planning, which allows the company to track the operational status of vessels (e.g., at sea, in port, under maintenance) and to respond promptly to any anomalies or issues.

Using the `status_id` as the surrogate key, we can efficiently query the fact table in conjunction with the status dimension table. Doing so will give the end-users more significant insights into the status of the ships of interest in our data warehouse. Having this relationship opens possibilities to more complex queries, such as tracking a ship's status history or a fleet of ships over time or analysing the ship statuses across different locations. Figure 4 shows the status dimension table after merging on status name and description. The full explanation of all the keys and measures in our status dimensions table can be found in Appendix A4.

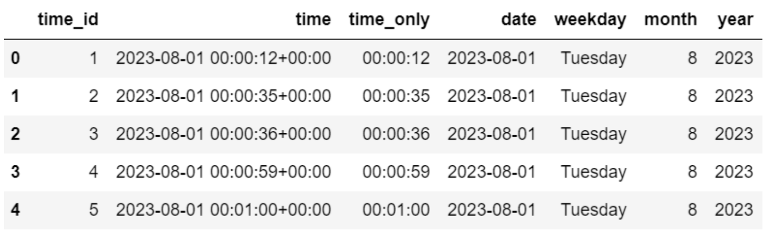


	status_id	status	status_name	description
0	1	0	Under way using engine	A vessel is considered to be underway when it ...
1	2	1	At anchor	A vessel is at anchor when it is held in posit...
2	3	2	Not under command	The term “not under command” means a vessel th...
3	4	3	Restricted Manoeuvrability	Manoeuvring characteristics include turning, y...
4	5	4	Constrained by draught	A power-driven vessel is severely restricted i...

Figure 4: Snippet of status dimensions table

2.3.4 Datetime Dimension Table

The datetime dimension is extracted using the time attribute in type 1 messages. The original time attribute comprises both date and time information for each vessel. For a detailed representation and analysis of the date and time dimension, we split the existing attribute into time, date, day of the week, month of the year, and year number. This splitting ensures greater freedom of querying vessel information based on a specified timeline. Figure 5 shows the datetime dimension table with `time_id` as a surrogate key to identify each combination of time split uniquely. This enhances our query search by linking the fact table with the date and time dimension table. Such flexibility allows companies to perform time-based analysis, such as identifying patterns over time, analysing performance by each day of the week, and predicting seasonal trends. The full explanation of all the keys and measures in our datetime dimensions table can be found in Appendix A5.



	time_id	time	time_only	date	weekday	month	year
0	1	2023-08-01 00:00:12+00:00	00:00:12	2023-08-01	Tuesday	8	2023
1	2	2023-08-01 00:00:35+00:00	00:00:35	2023-08-01	Tuesday	8	2023
2	3	2023-08-01 00:00:36+00:00	00:00:36	2023-08-01	Tuesday	8	2023
3	4	2023-08-01 00:00:59+00:00	00:00:59	2023-08-01	Tuesday	8	2023
4	5	2023-08-01 00:01:00+00:00	00:01:00	2023-08-01	Tuesday	8	2023

Figure 5: Sample of the datetime dimension table

2.3.5 Others

Apart from these dimensions, our group was also interested in including a weather dimension table but we were unsuccessful in scraping sufficient data for analysis to be meaningful. Please refer to Appendix B1 for more details and code.

3. Results

3.1 Data Presentation Area: Querying the Data (Q4)

This section presents the results of our data warehouse. As the basis for our analytical insights, the queries presented in this section are asked in the context of a maritime logistics company. This ensures that our data warehouse is relevant to a business context. As maritime logistics companies typically rely on precise tracking and detailed logistics to manage their fleet operations and route management, having access to historical data, such as ship movement, status, and traffic, is crucial for the company to make decisions.

Our group has prepared four queries to provide insights to the company:

3.1.1 How many unique ships of each name and type, registered under different countries, are present in the data. What is their last reported activity date, average speed, and how frequently do they report their status?

The query is designed to summarise the latest reported activities of ships from a fleet, including their types, total count of messages, and operation status. This query allows the company to monitor the most recent activities of their ship and understand which types of ships are in which locations and what is their operational status. Furthermore, this query helps the company understand fleet utilization and distribution by identifying unique ships associated with each activity. Finally, this query helps evaluate the frequency of communication from each ship, as indicated by the message count, which is essential for ensuring that ships report their positions as expected.

The SQL query assembles a summarised report from a maritime database, categorizing ships by name, type, registration country, and status. It provides a count of unique ships per category, the date of the last activity, their average speed, and the total number of communications for each group. The result is ordered to show the most recently active ships first, with a secondary sort by the fleet size in each category. This offers a snapshot of fleet operations, aiding in managing and strategically planning maritime activities. The SQL query and the output can be found in Appendix C1.

3.1.2 What is the average estimated fuel consumption of the different types of ships?

The rationale behind this query is because fuel is one of the most significant operational costs for a maritime logistics company, and estimating the fuel consumption costs will allow the company to manage better and forecast the costs, improving the company's profit position. Furthermore, understanding ships' fuel consumption from other fleets and companies and comparing it to the company's internal fleet can help monitor how fuel-efficient ships are. The ships that consume more fuel than the estimated ship type can then be flagged for maintenance.

In the absence of direct data on the fuel consumption of each ship in our data warehouse, we will need to estimate it. To approximate the fuel consumption, we use the following formula:

$$Fuel\ Consumption = efficiency\ index\ \times gt\ \times speed^2$$

Here, $Fuel\ Consumption$ is a function of the vessel's efficiency ($efficiency\ index$), gross tonnage (gt), and speed squared ($speed^2$). The $efficiency\ index$ is created in consultation with industry experts, where different types of ships will have different efficiencies when consuming fuel. For example, passenger ships and high-speed crafts will be more efficient due to their more streamlined hull form and is thus allocated a smaller $efficiency\ index$. For simplicity, we split the $efficiency\ index$ into three levels – high efficiency ($efficiency\ index = 0.8$), normal ($efficiency\ index = 1$), low efficiency ($efficiency\ index = 1.2$). gt is a proxy for the ship's size and capacity, while $speed^2$ encapsulates the non-linear relationship between the velocity of the ship and its fuel consumption.

Since the grain of our data warehouse is at the message level, we will be able to derive the above information by joining the ship dimension table to get gt, and the fact table to $speed$ at that point when the message is sent. The SQL query and the full output can be found in Appendix C2.

This query results are sorted by descending order of fuel consumption, and we can see that vehicles carriers and bulk carriers are the top ships with highest estimated fuel consumption, of which bulk carriers are part of the fleet of a maritime logistics company. With this information, the company can consider investing in ways to reduce the fuel consumption for the bulk carriers.

3.1.3 What is the message description of a specific ship at a specific date and time?

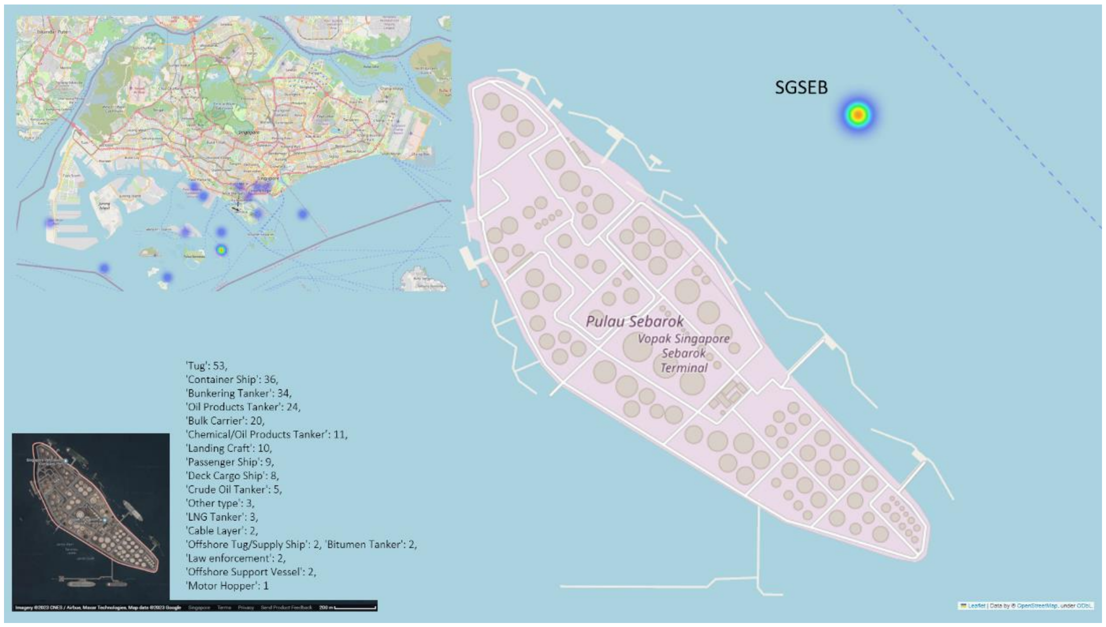
The findings from this query can describe the selected ship messages at a specific date and time. Maritime logistics companies can interact with the Python interface and select the specific ship name they want to view the messages. Maritime logistics companies can perform strategic, operational, and safety recommendations based on the messages. For instance, if an incident or abnormal event was reported at the given date and time, the query will allow the company to quickly glean insight into what occurred, allowing for thorough investigation and response. This query can also help monitor the performance of specific ships across time, checking if it is on schedule and identifying any deviations from the planned route. Having this function gives the company's team on land added control over their ships.

An example of the results is as follows: selecting the ship ‘RIGHTEOUS NN21’ (as shown in appendix C3 Output) outputs a message that is “A power-driven vessel is severely restricted in deviating from the course it follows because of the draught about the available depth and width of the navigable water,”. With this information, maritime logistics companies will be able to act and plan the vessel’s route carefully, considering the constraints. With this information, they can also inform port authorities and nearby vessels of the limited maneuverability. The SQL codes can be found in appendix C3: Query on detailed log from specific ship.

3.1.4 Which location in Singapore’s territorial waters is the most congested? Explain.

This query seeks to find the location that appears most frequently in AIS type 1 messages. Knowing the most congested areas allows for better strategic planning, allowing the company to anticipate potential delays and proactively manage scheduling for shipping routes. This information is critical in optimising route planning, which could save time and reduce costs associated with idling ships. This information can also be helpful in implementing better risk management strategies to prevent collisions and other maritime incidents, as congested areas have a higher risk of accidents.

The query identified 'SGSEB' as the primary source corresponding to Pulau Sebarok in Singapore, an island off the southern coast of Singapore. This is home to oil terminals and storage facilities, as shown in Fig. 6. This is unsurprising since Singapore is one of the world's busiest ports, particularly in the oil and gas sector. The frequent categorization of vessels as "Bunkering Tanker," "Oil Products Tanker," and "Chemical/Oil Products Tanker" suggests that the port activities in Sebarok are heavily geared towards the storage and transshipment of fuel and chemicals, which are consistent with the known industrial activities in that area. The tug-type vessel is the most common because it is essential for maneuvering vessels in and out of the crowded port area, assisting with berthing and emergency responses.

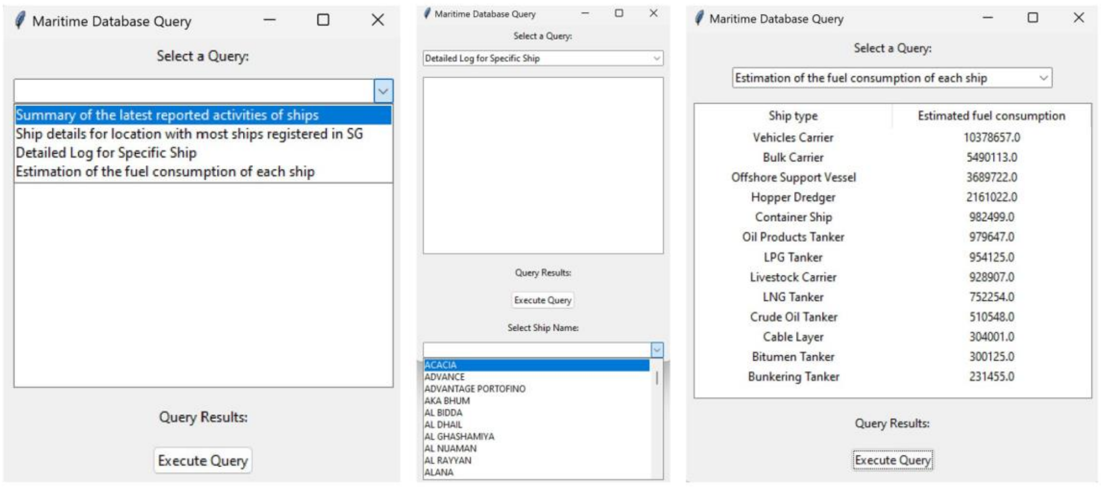


```
'Tug': 53,
'Container Ship': 36,
'Bunkering Tanker': 34,
'Oil Products Tanker': 24,
'Bulk Carrier': 20,
'Chemical/Oil Products Tanker': 11,
'Landing Craft': 10,
'Passenger Ship': 9,
'Deck Cargo Ship': 8,
'Crude Oil Tanker': 5,
'Other type': 3,
'LNG Tanker': 3,
'Cable Layer': 2,
'Offshore Tug/Supply Ship': 2, 'Bitumen Tanker': 2,
'Law enforcement': 2,
'Offshore Support Vessel': 2,
'Motor Hopper': 1
```

Figure 6: Location map from the Query

3.2 Data Access Tool: Querying Tool Interface (Q5)

We developed a user-friendly interface (UI) to enhance user engagement and streamline data retrieval. This UI acts as a conduit between the user and the PostgreSQL database using the psycopg2 package. Upon ensuring the installation of the necessary package requirements, users can activate the interface by running the provided script. The main window titled "Maritime Database Query" serves as the operational dashboard for data queries. Figure 7 shows the UI guide for step-by-step interactions.



Ship type	Estimated fuel consumption
Vehicles Carrier	10378657.0
Bulk Carrier	5490113.0
Offshore Support Vessel	3689722.0
Hopper Dredger	2161022.0
Container Ship	982499.0
Oil Products Tanker	979647.0
LPG Tanker	954125.0
Livestock Carrier	928907.0
LNG Tanker	752254.0
Crude Oil Tanker	510548.0
Cable Layer	304001.0
Bitumen Tanker	300125.0
Bunkering Tanker	231455.0

(a) Select a query (b) Provide Additional Input (c) Executing the query

Figure 7: User-friendly Interface (UI) guide

3.3 UI Guide (cont. Q5)

1. **Select a Query:**
- Initial View: Users are greeted by a dropdown box labeled "Select a Query" upon launching the application. This menu houses a compilation of predefined queries presented in Section 3.2.
- Query Selection: Users can browse and select the query that aligns with their informational needs.
2. **Provide Additional Input:**
- Conditional Dropdowns: For queries that require further specification, i.e., a particular ship, an additional dropdown labeled "Select Ship Name" will materialize post-query selection.
- Interactive Selection: Users can select the desired vessel to provide an interactive approach to allow users to have the flexibility to view selected ships.
3. **Executing the query**
- Activation: With the query and any additional parameters set, initiate the data fetch by clicking "Execute Query."
- Results Display: The results will populate the main window, with the columnar arrangement adapting to the nature of the executed query.
4. **Closing the application**
- Termination: Terminate the application using the standard window close control "x."

4. Conclusion

In conclusion, this report has demonstrated the vital role of a structured data warehouse in harnessing AIS messages to enhance maritime navigation and safety. Firstly, we presented the process of selecting a suitable timeframe for the study. We conducted EDA on the dataset before proposing and designing our star schema. We pre-processed type 1 and type 5 messages and performed web-scraping to populate missing values. Finally, we conducted data transformation and stored the data on PostgreSQL. Results from the executed queries suggest that our established data warehouse can provide actionable insights to enhance decision-making capabilities. Type 1 and type 5 messages provided a comprehensive navigational and vessel-specific information repository. Future work can include developing a graphics interface, advanced analytical techniques such as machine learning, and exploring predictive modeling. These advancements could provide deeper insights into maritime operations and prescriptive analytics, such as forecasting potential logistics disruption.

Appendix:

- Failed attempts: Scraping weather data
- Full explanation of fact and dimension tables
- Additional queries

Appendix A: Details on Fact and Dimension Tables

A1: Detailed Explanation of the Fact Table

Our fact table consists of surrogate keys that allow the fact table to join to the respective dimension tables, and the measures.

- **`message_id`**: This is a unique identifier for each record in the table. In the context of a fact table, it's the primary key that uniquely identifies each row.
- **`location_id`**: Represents a unique identifier associated with a geographic location. In the star schema, this acts as the foreign key linked to a location dimension table that provides more details about the location.
- **`ship_id`**: Identifies the specific ship to which the message pertains. This links to the ship dimension table where further details of each ship, such as the name, type, size, callsign etc. are stored.
- **`time_id`**: A reference to the specific time at which the event (the sending of the message) occurred. It links to the time dimension table where each time_id corresponds to more granular time data such as the date, week, month, year etc.
- **`status_id`**: corresponds to a status code or identifier that describes the condition or state of the ship at the time the message was sent. This acts as the foreign to link the status dimension table.
- **`accuracy`**: flag that indicates the accuracy of the position fix, with 1 indicating a high accuracy fix and 0 indicating a lower accuracy fix. In navigational terms, this could reflect the confidence in the positional data due to factors like GPS signal quality.
- **`course`**: The intended navigational path of the ship, in degrees, where 0 degrees represents true north. The course is typically set by the ship's navigation system and may differ from the heading due to environmental factors or the ship's current manoeuvres.
- **`heading`**: The actual direction the ship's bow is pointing, also in degrees. Heading can be affected by wind, current, and the ship's own motion and is an essential factor in navigation to ensure the ship follows its intended course.
- **`speed`**: The speed at which the ship is traveling. This is often measured in knots (one nautical mile per hour). Speed is a crucial factor in calculating arrival times and in maintaining safe operations at sea.
- **`turn`**: This represents the rate at which the ship is turning or the angle of a turn. A positive value could indicate a turn to the right (starboard), and a negative value a turn to the left (port).

A2: Detailed Explanation of the Location Dimension Table

- **location_id**: This is a unique identifier for each location entry within the dimension table. It links event or fact to a specific location.
- **ship_lon**: This represents the longitude coordinate of the location. Longitude is a geographic coordinate that specifies the east-west position of a point on the Earth's surface. It is usually expressed in degrees, minutes, and seconds, or in decimal form, as seen here.
- **ship_lat**: This is the latitude coordinate of the location. Latitude is a geographic coordinate that specifies the north-south position of a point on the Earth's surface. Like longitude, it is usually expressed in degrees, minutes, and seconds, or in decimal form.
- **LOCODE**: location code is a universal code based on the UNECE dataset. They can be modified to include other LOCODE not included in the dataset.
- **portname:** name of the port, data from vesselfinder
- **Country:** country of the port, data from vesselfinder
- **Coordinates:** DD coordinates, data from both vesselfinder and UN data, converted to DD
- **loc_lat:** latitude location of the mapped LOCODE
- **loc_lon:** longitude location of the mapped LOCODE

A3: Detailed Explanation of the Ship Dimension Table

- **ship_id**: A unique identifier for each ship within the data warehouse. It's used as a primary key within this dimension table and as a foreign key in the fact table to link the dimensions to the facts.
- **mmsi**: The Maritime Mobile Service Identity, which is a unique 9-digit number that identifies a ship's radio and is used worldwide for maritime communication. It's crucial for tracking and identifying vessels in the AIS.
- **imo**: International Maritime Organization number, which is a unique seven-digit number assigned to seafaring vessels. It's permanent and remains associated with the ship throughout its lifetime, regardless of name or flag changes.
- **callsign**: The unique radio call sign assigned to a ship. This alphanumeric code is used to identify the ship during voice communications over the radio.
- **shipname**: The name of the ship. This is the common name used to identify the vessel and is typically displayed on the hull.
- **country:** country of origin or where its built.
- **shiptype**: A code representing the type of ship. This could be linked to a reference table that explains the type, such as cargo, tanker, passenger, etc.
- **to_bow**: The distance from the AIS transponder to the bow of the ship. This measurement is essential for calculating the exact position of the ship's front end.
- **to_stern**: The distance from the AIS transponder to the stern (back end) of the ship. Similar to 'to_bow', it's used to pinpoint the ship's position.
- **to_port**: The distance from the AIS transponder to the port side (left side when facing the bow) of the ship.
- **to_starboard**: The distance from the AIS transponder to the starboard side (right side when facing the bow) of the ship.

A4: Detailed Explanation of the Status Dimension Table

- **status_id**: This is a unique identifier for each status entry in the dimension table. It associates status information with the recorded facts.
- **status**: This is a numeric code or an enumeration representing the different statuses a vessel can report.
- **status_name**: This is a human-readable name that corresponds to the status code. It provides an easily understood description of the vessel's status, such as "Under way using engine", "At anchor", etc.
- **description**: This field contains a detailed explanation of what each status means. It provides context that can be very important for analysis, such as understanding the vessel's operational mode, any restrictions it might be under, or specific conditions affecting its navigation.

A5: Detailed Explanation of the DateTime Dimension Table

- **`time_id`**: This is a unique identifier for each time entry and serves as the primary key of the dimension table. It is used to join the dimension table with fact tables that have a temporal component.
- **`time_only`**: Contains the time of day for the event without a date component in the format of HH:MM:SS. This could be used for analyzing events that happen at specific times of the day, irrespective of the date.
- **`date`**: The date on which the event occurred in the format of DD-MM-YYYY. This allows for analysis by date and is useful for tracking trends over time.
- **`weekday`**: The day of the week for the corresponding date. This is useful for analysis that requires understanding patterns by day of the week, such as increased activity on weekdays versus weekends.
- **`month`**: The month of the year for the date. This column is useful for performing monthly trend analysis and seasonal comparisons.
- **`year`**: The year of the event. This enables year-over-year analyses and helps in identifying long-term trends.

Appendix B1: Unsuccessful scraping of weather data

While we were able to scrape some data, we were unable to scrape enough information from free sources to make any meaningful analyses. Our code to scrape the data is the following:

```
import requests
import pandas as pd

df = pd.read_csv("small_location_data.csv")
location_data = df.head(25)

import requests

API_KEY = 'hashed_for_privacy'

def build_api_url(lon, lat, time, api_key):
    """
    Constructs the API request URL for a given set of longitude, latitude, and time.

    :param lon: Longitude of the location
    :param lat: Latitude of the location
    :param time: Date and time for the weather data
    :param api_key: User's API key for the Visual Crossing Weather API
    :return: API request URL as a string
    """
    base_url                                                                                         =
"https://weather.visualcrossing.com/VisualCrossingWebServices/rest/services/timeline/"
    # Format the time to only include the date part (yyyy-MM-dd)
    time_formatted = time.split('T')[0]
    # Construct the URL
    api_url = f"{base_url}{lat},{lon}/{time_formatted}?key={api_key}&include=hours"
    return api_url

def get_weather_data(api_url):
    """
    Makes an API call to the given URL and returns the weather data.

    :param api_url: The full API URL to make the request to
    :return: Parsed weather data as a dictionary
    """
    try:
        response = requests.get(api_url)
        # Check if the call was successful
        if response.status_code == 200:
            weather_data = response.json()
            return weather_data
        else:
            print(f"Error: {response.status_code}")
        return None
    except Exception as e:
        print(f"An error occurred: {e}")
        return None

# This function will be used to loop over the entire dataset and make the API calls.
def fetch_weather_data_for_all_locations(df, api_key):
    weather_data_list = []  # List to store all weather data
    failed_requests = []  # List to store any locations for which the API call fails

    for index, row in df.iterrows():
        # Build the API URL for the current row's data
        api_url = build_api_url(row['lon'], row['lat'], row['time'], api_key)
        print(f"Fetching data for index {index}: {api_url}")  # Print the URL being called

        # Make the API call and store the weather data
        weather_data = get_weather_data(api_url)

        if weather_data:
            # Add the successful result to the list (commented out since we can't call the API)
            weather_data_list.append(weather_data)
        else:
            # Log the failed request for later inspection
            failed_requests.append((index, row['lon'], row['lat'], row['time']))

    return weather_data_list, failed_requests
```

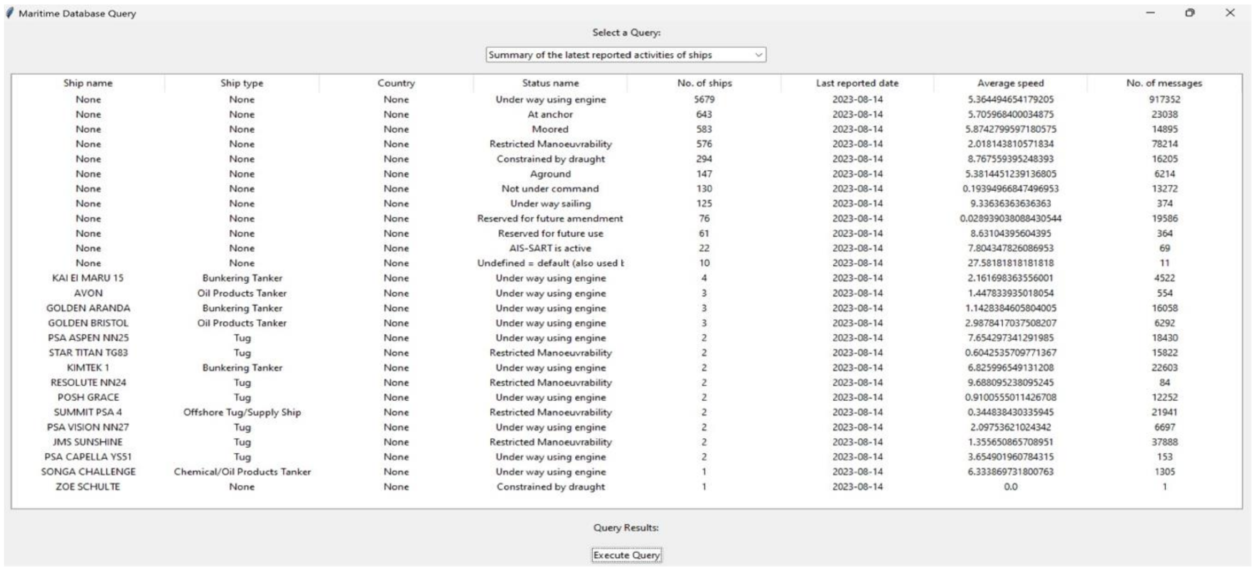
Appendix C: SQL Queries

C1: SQL Query for general outlook of ocean activity

```
SELECT
    s.shipname, s.shiptype_name, l.country, st.status_name,
    COUNT(DISTINCT s.ship_id) AS number_of_ships,
    MAX(d.date) AS last_reported_date,
    AVG(m.speed) AS average_speed,
    COUNT(m.message_id) AS message_count
FROM
    message_fact m, ship_dim s, location_dim l, datetime_dim d, status_dim st

WHERE
    m.ship_id = s.ship_id AND
    m.location_id = l.location_id AND
    m.datetime_id = d.time_id AND
    m.status_id = st.status_id
GROUP BY
    s.shipname,
    s.shiptype_name,
    l.country,
    st.status_name
ORDER BY
    last_reported_date DESC,
    number_of_ships DESC;
```

C1: Output

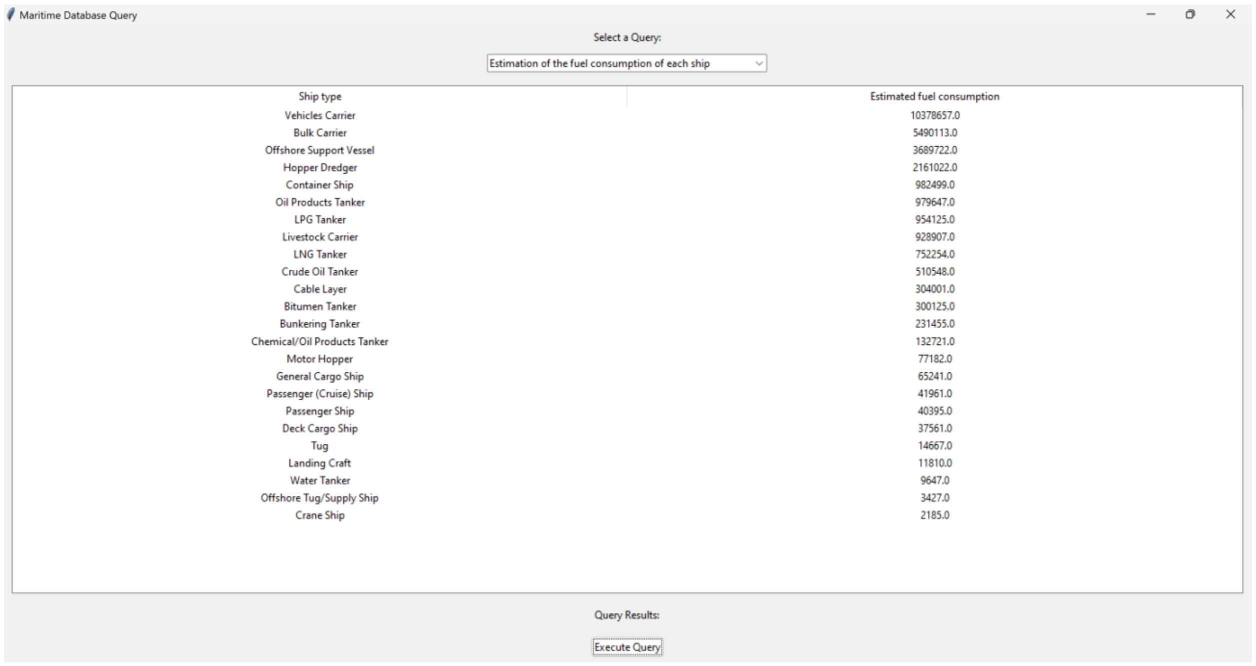


Ship name	Ship type	Country	Status name	No. of ships	Last reported date	Average speed	No. of messages
None	None	None	Under way using engine	5679	2023-08-14	5.364494654179205	917352
None	None	None	At anchor	643	2023-08-14	5.705968400034875	23038
None	None	None	Moored	583	2023-08-14	5.8742799597180575	14895
None	None	None	Restricted Manoeuvrability	576	2023-08-14	2.018143810571834	78214
None	None	None	Constrained by draught	294	2023-08-14	8.767559395248393	16205
None	None	None	Aground	147	2023-08-14	5.3814451239136805	6214
None	None	None	Not under command	130	2023-08-14	0.19394966847496953	13272
None	None	None	Under way sailing	125	2023-08-14	9.33636363636363	374
None	None	None	Reserved for future amendment	76	2023-08-14	0.028939038088430544	19586
None	None	None	Reserved for future use	61	2023-08-14	8.63104395604395	364
None	None	None	AIS-SART is active	22	2023-08-14	7.804347826086953	69
None	None	None	Undefined = default (also used t	10	2023-08-14	27.581818181818	11
KAI EI MARU 15	Bunkering Tanker	None	Under way using engine	4	2023-08-14	2.161698363556001	4522
AVON	Oil Products Tanker	None	Under way using engine	3	2023-08-14	1.447833935018054	554
GOLDEN ARANDA	Bunkering Tanker	None	Under way using engine	3	2023-08-14	1.1428384605804005	16058
GOLDEN BRISTOL	Oil Products Tanker	None	Under way using engine	3	2023-08-14	2.9878417037508207	6292
PSA ASPEN NN25	Tug	None	Under way using engine	2	2023-08-14	7.654297341291985	18430
STAR TITAN TG83	Tug	None	Restricted Manoeuvrability	2	2023-08-14	0.6042535709771367	15822
KIMTEK 1	Bunkering Tanker	None	Under way using engine	2	2023-08-14	6.825996549131208	22603
RESOLUTE NN24	Tug	None	Restricted Manoeuvrability	2	2023-08-14	9.688095238095245	84
POSH GRACE	Tug	None	Under way using engine	2	2023-08-14	0.9100555011426708	12252
SUMMIT PSA 4	Offshore Tug/Supply Ship	None	Restricted Manoeuvrability	2	2023-08-14	0.344838430335945	21941
PSA VISION NN27	Tug	None	Under way using engine	2	2023-08-14	2.09753621024342	6697
JMS SUNSHINE	Tug	None	Restricted Manoeuvrability	2	2023-08-14	1.355650865708951	37888
PSA CAPELLA YS51	Tug	None	Under way using engine	2	2023-08-14	3.654901960784315	153
SONGA CHALLENGE	Chemical/Oil Products Tanker	None	Under way using engine	1	2023-08-14	6.333869731800763	1305
ZOE SCHULTE	None	None	Constrained by draught	1	2023-08-14	0.0	1

C2: SQL Query for Fuel Consumption by ship type

```
SELECT
    s.shiptype_name,
    ROUND(AVG(
        CASE
            WHEN LEFT(s.shiptype, 1) IN ('4', '6') THEN 0.8 * s.gt * (m.speed * m.speed)
            WHEN LEFT(s.shiptype, 1) IN ('7', '8') THEN 1.2 * s.gt * (m.speed * m.speed)
            ELSE s.gt * (m.speed * m.speed)
        END)
    ) AS estimated_fuel_consumption
FROM ship_dim s, message_fact m
WHERE s.ship_id = m.ship_id
AND s.shiptype_name IS NOT NULL
AND s.gt IS NOT NULL
GROUP BY s.shiptype_name
HAVING AVG(m.speed) > 0
ORDER BY estimated_fuel_consumption DESC;
```

C2: Output

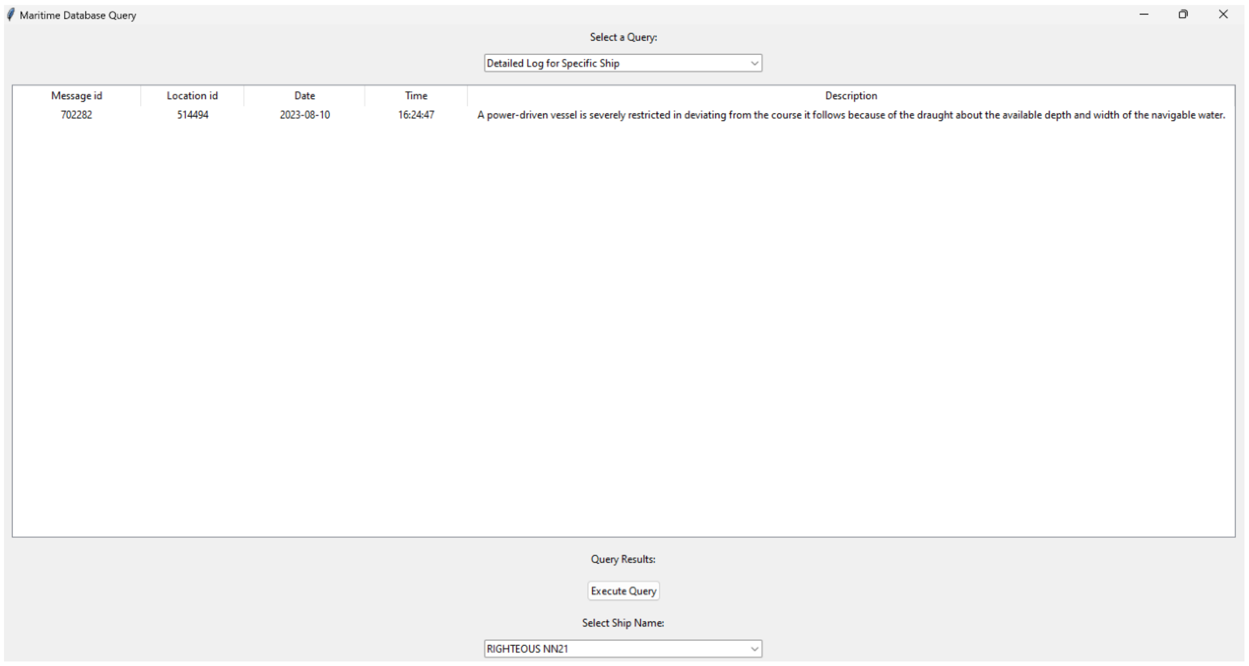


Ship type	Estimated fuel consumption
Vehicles Carrier	10378657.0
Bulk Carrier	5490113.0
Offshore Support Vessel	3689722.0
Hopper Dredger	2161022.0
Container Ship	982499.0
Oil Products Tanker	979647.0
LPG Tanker	954125.0
Livestock Carrier	928907.0
LNG Tanker	752254.0
Crude Oil Tanker	510548.0
Cable Layer	304001.0
Bitumen Tanker	300125.0
Bunkering Tanker	231455.0
Chemical/Oil Products Tanker	132721.0
Motor Hopper	77182.0
General Cargo Ship	65241.0
Passenger (Cruise) Ship	41961.0
Passenger Ship	40395.0
Deck Cargo Ship	37561.0
Tug	14667.0
Landing Craft	11810.0
Water Tanker	9647.0
Offshore Tug/Supply Ship	3427.0
Crane Ship	2185.0

C3: Query on detailed log for specific ship

```
SELECT
        mf.message_id,
        mf.location_id,
        dt.date,
        dt.time_only as time,
        s.description
FROM
        message_fact mf
JOIN
        status_dim s ON mf.status_id = s.status_id
JOIN
        datetime_dim dt ON mf.datetime_id = dt.time_id
JOIN
        ship_dim sd ON mf.ship_id = sd.ship_id
WHERE
         sd.shipname = '{ship_name}'
ORDER BY
         dt.date, dt.time_only;
```

C3 Output



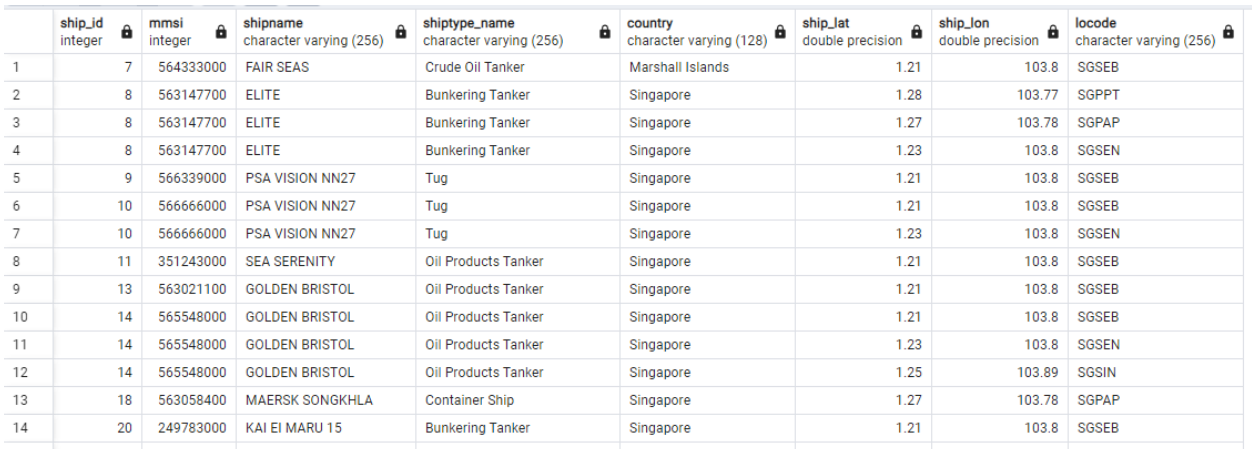
Message id	Location id	Date	Time	Description
702282	514494	2023-08-10	16:24:47	A power-driven vessel is severely restricted in deviating from the course it follows because of the draught about the available depth and width of the navigable water.

C4: Query

```
WITH uniqueships AS (
   SELECT  mf.ship_id, ld.ship_lon, ld.ship_lat, ld.locode
   FROM message_fact mf
   INNER JOIN location_dim ld ON mf.location_id = ld.location_id
   WHERE ld.LOCODE LIKE 'SG%'
   GROUP BY mf.ship_id, ld.ship_lon, ld.ship_lat, ld.locode)

SELECT
   sd.ship_id, sd.mmsi, sd.shipname, sd.shiptype_name, sd.country,
   usg.ship_lat, usg.ship_lon, usg.locode
FROM ship_dim sd
JOIN uniqueships usg ON sd.ship_id = usg.ship_id
ORDER BY sd.ship_id;
```

C4: Output



	ship_id integer	mmsi integer	shipname character varying (256)	shiptype_name character varying (256)	country character varying (128)	ship_lat double precision	ship_lon double precision	locode character varying (256)
1	7	564333000	FAIR SEAS	Crude Oil Tanker	Marshall Islands	1.21	103.8	SGSEB
2	8	563147700	ELITE	Bunkering Tanker	Singapore	1.28	103.77	SGPPT
3	8	563147700	ELITE	Bunkering Tanker	Singapore	1.27	103.78	SGPAP
4	8	563147700	ELITE	Bunkering Tanker	Singapore	1.23	103.8	SGSEN
5	9	566339000	PSA VISION NN27	Tug	Singapore	1.21	103.8	SGSEB
6	10	566666000	PSA VISION NN27	Tug	Singapore	1.21	103.8	SGSEB
7	10	566666000	PSA VISION NN27	Tug	Singapore	1.23	103.8	SGSEN
8	11	351243000	SEA SERENITY	Oil Products Tanker	Singapore	1.21	103.8	SGSEB
9	13	563021100	GOLDEN BRISTOL	Oil Products Tanker	Singapore	1.21	103.8	SGSEB
10	14	565548000	GOLDEN BRISTOL	Oil Products Tanker	Singapore	1.21	103.8	SGSEB
11	14	565548000	GOLDEN BRISTOL	Oil Products Tanker	Singapore	1.23	103.8	SGSEN
12	14	565548000	GOLDEN BRISTOL	Oil Products Tanker	Singapore	1.25	103.89	SGSIN
13	18	563058400	MAERSK SONGKHLA	Container Ship	Singapore	1.27	103.78	SGPAP
14	20	249783000	KAI EI MARU 15	Bunkering Tanker	Singapore	1.21	103.8	SGSEB