Relational Database for SafeMarine

Mahdsiah Khaalik
Victor Lam
M. Furkan Oruc
Bhargav Nelluri

12.13.2021

Table of Contents

1. System Requirements

SafeMarine is a maritime technology startup, specializing in prevention of potential maritime accidents in narrow and congested waterways. SafeMarine provides spatio-temporal data based analytics solutions to their customers. They rely heavily on maritime data and their data analytics processes require specialized relational database designs, specific for their analytics purposes.

Primary data source utilized by SafeMarine is Automatic Identification System (AIS) data. AIS messages are transmitted by marine vessels and they include static, dynamic and voyage related information. AIS is frequently used by maritime authorities around the world to obtain live information on vessels' identity, position, length, speed, estimated arrival time to the port and others. Since AIS is transmitted and stored in a mass and unstructured way, the raw version is not useful for analytics purposes. In the scope of this technical report, a relational database management system design is presented for SafeMarine's business purposes.

1.1. Client Requirements

1.1.1. Each ship transmits regular messages about their real time information. Ships are uniquely identified with their message_Id. Real time information includes their speed, location and other real time data. (DYNAMIC) These messages can not be present without a ship it belongs to. These are provided as:

- velocity (Speed),
- longitude (Longitude),
- latitude (Latitude),
- heading (Heading),
- exact time for delivery of the message data (Timestamp),
- estimated arrival location (Destination),
- estimated arrival time (Eta),
- message accuracy (Accuracy),
- vessel's navigational status (status),
- device ID on the owner ship (device_Id),
- device accuracy (Device_Accuracy)

1.1.2. Ships are the most crucial element of maritime traffic (SHIP). Uniquely identified with MMSI_Id. Each ship can be inside a certain port area, waterway or can be present in the open sea. Each ship has:

- a unique identifier code (Mmsi_Id),
- international maritime organization code (Imo),
- a name (Name),

- length (Length),
- width (Width),
- type of ship, as passenger ship, cargo ship, tanker or etc. (Ship_type),
- message transmitter device type (Device).

1.1.3. Ships pass through canals, waterways and ports through their voyage. (WATERWAY) Each canal, waterway have characteristics and they are uniquely identified by their name. These information are stored as:

- name of the area (Name),
- capacity (Capacity).

☐ To determine the area location:

- maximum latitude (Max_latitude),
- maximum longitude (Max_longitude),
- minimum latitude (Min_latitude),
- minimum longitude (Min_longitude)

1.1.4. Ships dock at ports throughout their journeys. They are uniquely identified by their name. Ports (PORT) are represented as:

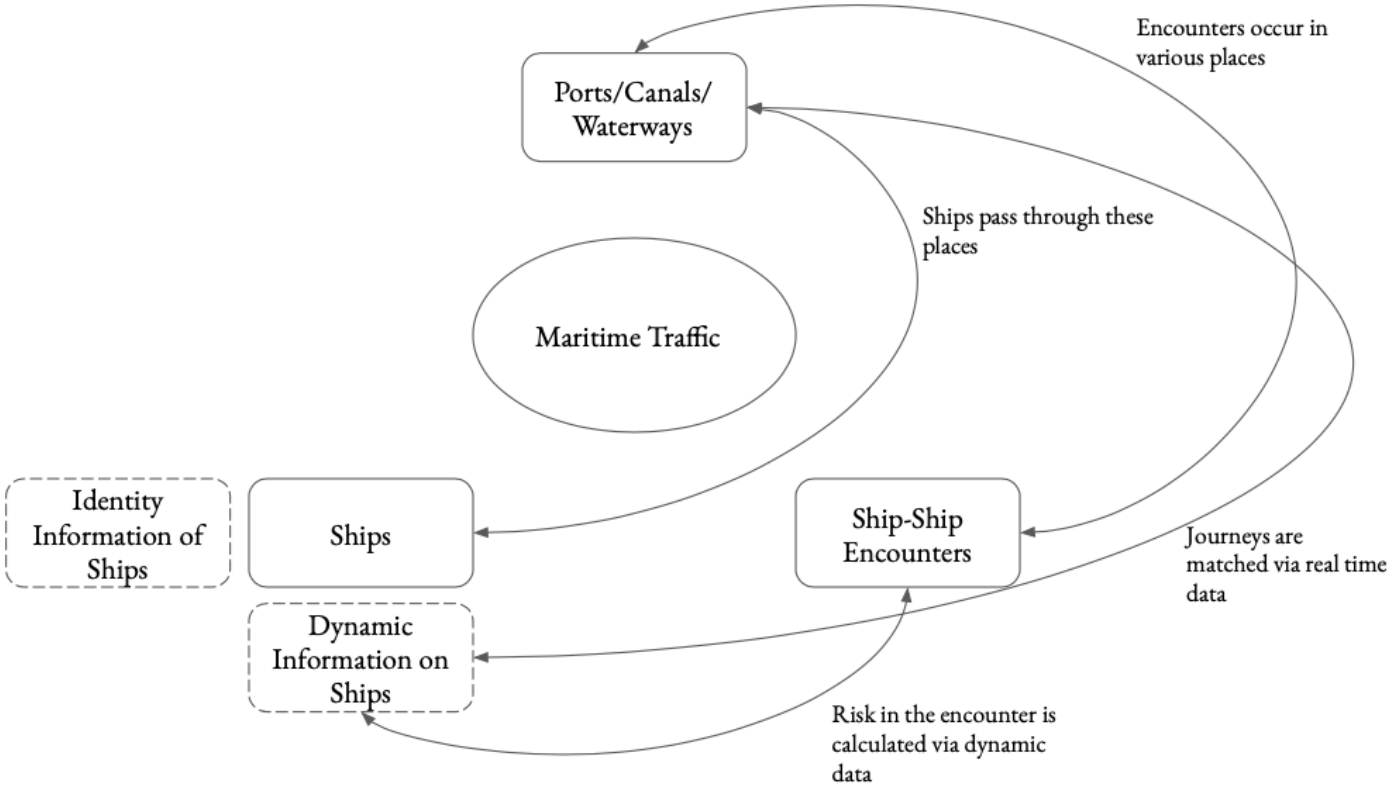
- name of the port (Name),
- capacity (Capacity).

☐ To determine the area location:

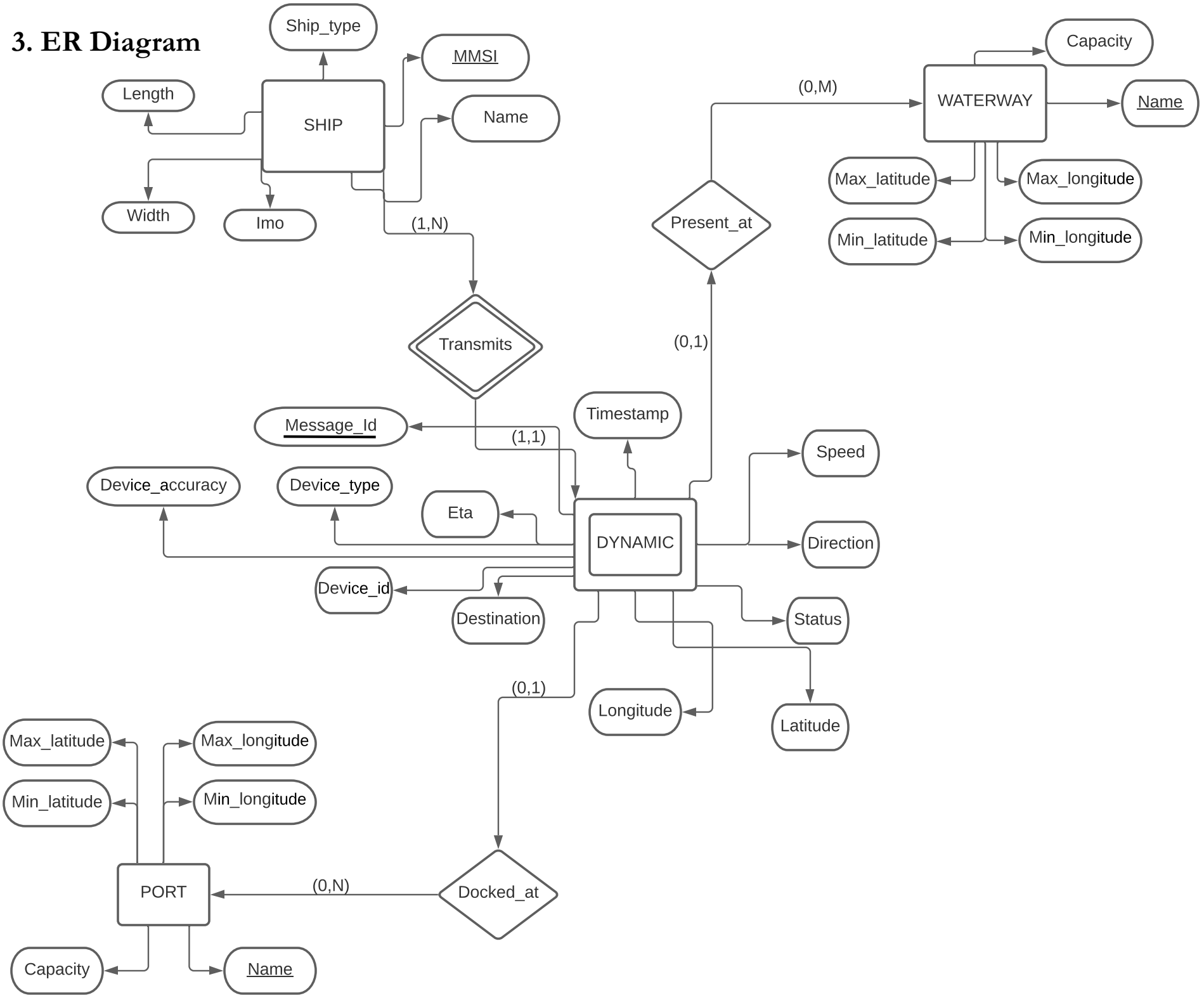
- maximum latitude (Max_latitude),
- maximum longitude (Max_longitude),
- minimum latitude (Min_latitude),
- minimum longitude (Min_longitude)

❖ Waterways and ports are distinctly defined since they have different restrictions and they may need to be defined with different attributes in the future due to their requirements.

2. Contextual Data Flow Diagram

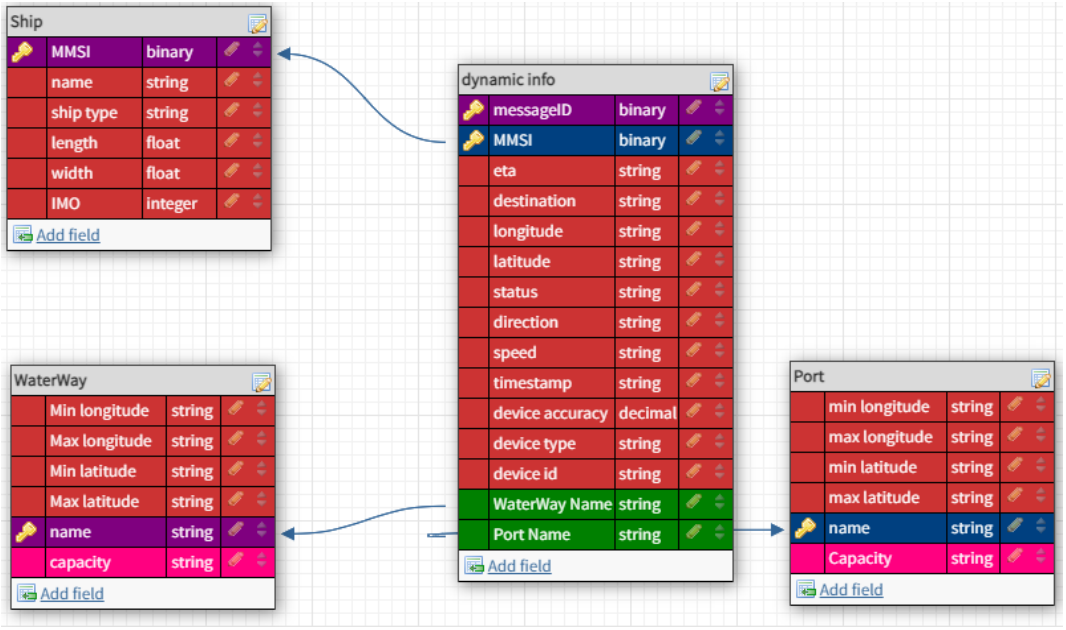


3. ER Diagram

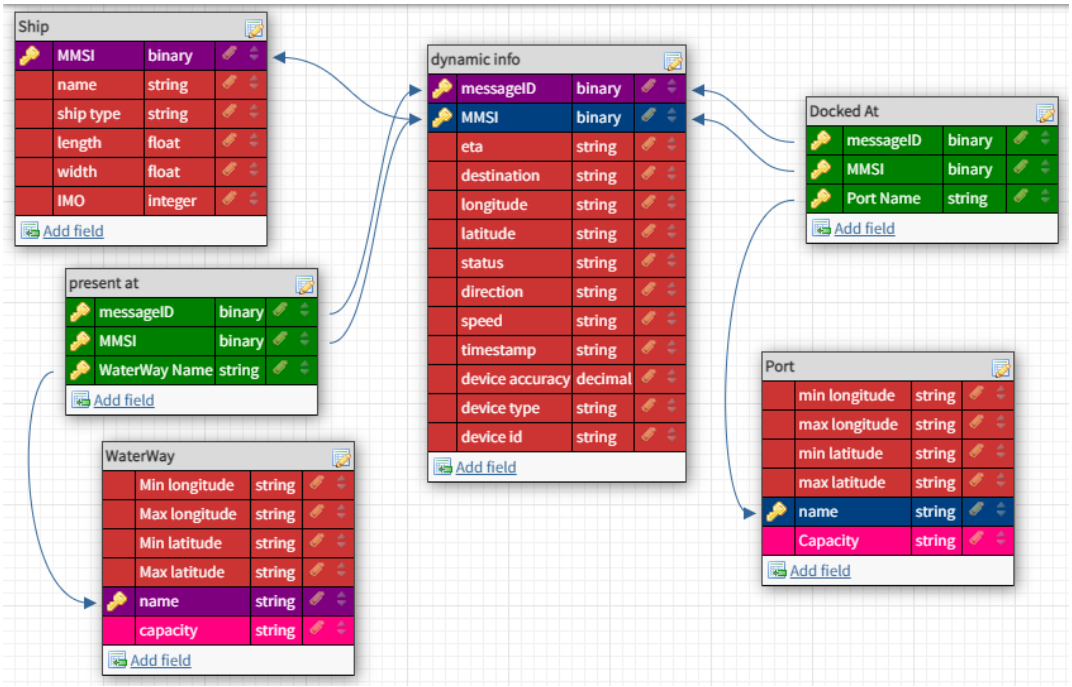

Ship_type
MMSI
Length
SHIP
Name
Width
Imo
(1,N)
Transmits
Message_Id
Timestamp
(1,1)
Device_accuracy
Device_type
Eta
DYNAMIC
Speed
Direction
Device_id
Destination
Status
Longitude
Latitude
(0,1)
Present_at
(0,M)
WATERWAY
Capacity
Name
Max_latitude
Max_longitude
Min_latitude
Min_longitude
Max_latitude
Max_longitude
Min_latitude
Min_longitude
PORT
(0,N)
Docked_at
Capacity
Name

4. Normalized DB Model

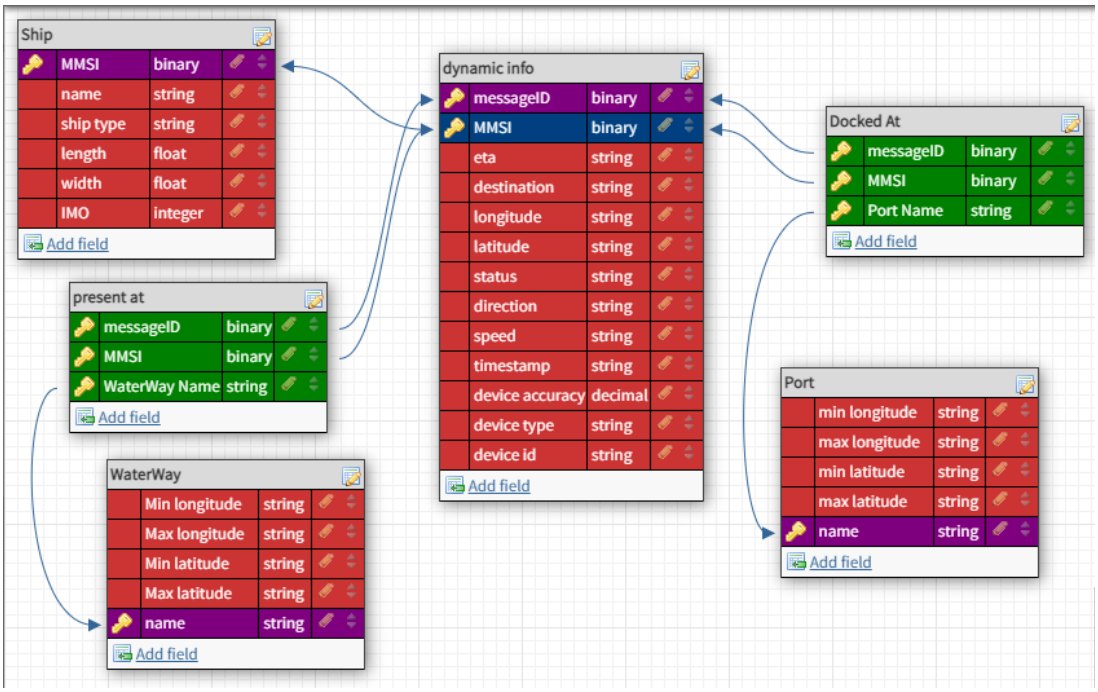
(1NF)



(2NF)



(3NF)


Ship
MMSI binary
name string
ship type string
length float
width float
IMO integer
Add field
dynamic info
messageID binary
MMSI binary
eta string
destination string
longitude string
latitude string
status string
direction string
speed string
timestamp string
device accuracy decimal
device type string
device id string
Add field
Docked At
messageID binary
MMSI binary
Port Name string
Add field
present at
messageID binary
MMSI binary
WaterWay Name string
Add field
WaterWay
Min longitude string
Max longitude string
Min latitude string
Max latitude string
name string
Add field
Port
min longitude string
max longitude string
min latitude string
max latitude string
name string
Add field

5. Recommendation on DBMS

We have considered MYSQL, mongo DB, PostgreSQL, and oracle DB for executing our project. Below we have listed pros and cons for each Database program that we wanted to use.

MYSQL

Pros

- MySQL is an open-source database. Therefore, it is free to use. It is stable and reliable software.
- MySQL is known for being the most secure and reliable database management system.
- MySQL offers unmatched scalability to facilitate the management of apps using a smaller footprint even in companies having large amounts of data.
- MySQL has features like complete atomic, consistent, isolated, durable transaction support, multi-version transaction support, and unrestricted row-level locking.
- It is supported by all the available platforms (windows, Linux etc.)
- MySQL is known for being fast and reliable. It is fast in retrieving data from the database.

Cons

- MySQL doesn't support huge databases efficiently.
- Sometimes MySQL suffers from stability issues.
- It sometimes suffers from stability issues.

MongoDB

Pros

- It is also free & open source to use, but its design principles are entirely different from the traditional relational systems.
- MongoDB adopts a different approach to storing data, and the information is represented in the form of a series of JSON documents.
- It is a more flexible approach because documents are self-describing and have attributes like replication and gridFS (specification for storing & retrieving, it divides the large file into parts).
- It uses sharding while handling large datasets to distribute the data to multiple servers.

Cons

- In MongoDB joining documents is a tedious task because it fails to support joins as relational DB. Even though we can utilize joins functionality by adding the code manually, it consumes more time.
- Another problem is indexing; if implemented incorrectly, the performance is deficient
- It allows a limited size document of only 16MB, so the performance nesting for all documents is limited.
- Another problem is duplication of data; it makes it challenging to handle the data sets if relations are not defined well, which may lead to data corruption.
- It uses a high amount of storage due to a lack of joins functionalities and increases in data redundancy.
- We use MySQL because MongoDB fails the relational DB, and our project mainly focuses on the relational DB, and our data is structured, so that's why we used MySQL.

PostgreSQL

Pros

- The vital aspect of PostgreSQL is it allows Query Parallelism.
- It is a feature-rich database that can handle complex queries and massive databases & massive databases.
- Another important thing is it has a strong adoption of most of the tools and works well with them; the ODBC & JDBC connections work well.

Cons

- It doesn't provide any data compression, which limits the data ingestion velocity, and performance is low.
- PostgreSQL mainly focuses on compatibility.
- On Performance metrics, it is slower compared to MySQL.
- It is flexible for large data volumes, and configuration can be confusing, so our project consists of a small volume of data, and MySQL is easy to use and process.

Oracle DB

Pros

- These are used for many corporation-level applications.

- These databases are highly reliable & deliver excellent performance and try to provide high integrity of data storage.

Cons

- The Oracle product can be several times more expensive than the MySQL option.
- It is more complex compared to MySQL & requires great skills to install and maintain it due to the complex engine.
- Both MySQL and Oracle SQL are RDBMS, but our data is in small volume, and MySQL handles it very efficiently.

Considering all the pros and cons of all the databases in consideration, we have chosen MySQL for implementing our project because of its ease of implementation and our team members previous experience on working with MySQL.

6. Implementation - Ready DB Model

We used the first normalization because it requires less space when reflecting relationships, and makes joins simpler.

```
drop table if exists DYNAMIC;

drop table if exists SHIP;

drop table if exists WATERWAY;

drop table if exists PORT;


 create table SHIP(

MMSI float(10),

NAME varchar(20),

SHIP_TYPE VARCHAR(10),

LENGTH float(10) CHECK (LENGTH>0),

WIDTH float(10) CHECK (WIDTH>0),

IMO float(10) not null,

primary key (MMSI)

);


 create table WATERWAY(

MAX_LONGITUDE varchar(10),

MAX_LATITUDE VARCHAR(10),

MIN_LONGITUDE varchar(10),

MIN_LATITUDE VARCHAR(10),
```

```
NAME VARCHAR(20),
Capacity float(10),
primary key (NAME)
);

create table PORT(
MAX_LONGITUDE varchar(10),
MAX_LATITUDE VARCHAR(10),
MIN_LONGITUDE varchar(10),
MIN_LATITUDE VARCHAR(10),
NAME VARCHAR(20),
Capacity float(10),
primary key (NAME)
);

create table DYNAMIC(
SHIP_MMSI float(10),
MESSAGE_ID binary(20),
PORT_NAME VARCHAR(20),
WATERWAY_NAME VARCHAR(20),
TIME_STAMP timestamp,
SPEED varchar(10),
DIRECTION varchar(10),
```

```
STATUS varchar(10),

LONGITUDE varchar(10),

LATITUDE varchar(10),

DESTINATION varchar(20),

ETA datetime,

DEVICE_ACCURACY varchar(5),

DEVICE_TYPE varchar(10),

primary key (MESSAGE_ID),

foreign key (WaterWay_Name) references WATERWAY(NAME),

foreign key (Port_Name) references PORT(NAME)

);


INSERT INTO SHIP VALUES('11873', 'EVERGIVEN', 'COMMERCIAL', '280', '85',
'3424782');

INSERT INTO SHIP VALUES('16930', 'QUEEN MARY', 'TANKER', '350', '125', '2342356');

INSERT INTO SHIP VALUES('18462', 'ECLIPSE', 'COMMERCIAL', '420', '140', '2112557');

INSERT INTO SHIP VALUES('49539', 'QUEEN ELIZABETH', 'TANKER', '190', '65',
'3463452');

INSERT INTO SHIP VALUES('56924', 'OASIS OF THE SEAS', 'COMMERCIAL', '310', '170',
'234264');

INSERT INTO SHIP VALUES('49068', 'EMMA MAERSK', 'COMMERCIAL', '315', '145',
'64756124');

INSERT INTO SHIP VALUES('19232', 'CUMHURIYET', 'PRIVATE', '65', '14', '19231881');
```

INSERT INTO WATERWAY VALUES('29.35', '41.22', '27.63', '39.46', 'STRAIT OF ISTANBUL', '45');

INSERT INTO WATERWAY VALUES('11.26', '57.53', '9.45', '54.39', 'KATTEGAT STRAIT', '200');

INSERT INTO WATERWAY VALUES('0.1', '0.0', '0.0', '0.0', 'ABSENT', '0');

INSERT INTO PORT VALUES('4.40', '51.24', '4.38', '51.15', 'PORT OF ANTWERP', '20');

INSERT INTO PORT VALUES('0.0', '0.0', '0.0', '0.0', 'ABSENT', '0');

INSERT INTO PORT VALUES('103.58', '1.13', '103.20', '1.03', 'PORT OF SINGAPORE', '45');

INSERT INTO DYNAMIC VALUES('11873', '129475',

'PORT OF ANTWERP', 'ABSENT', '2021-04-07 04:10:05', '5', 'East', 'True',

'4.40', '51.21', 'SHANGAI', '2021-05-01 03:09:16', 'False', 'MOBILE');

INSERT INTO DYNAMIC VALUES('16930', '2436456',

'ABSENT', 'KATTEGAT STRAIT', '2020-02-05 01:11:05', '5', 'West', 'True',

'11.26', '57.53', 'GEORGIA', '2021-05-01 04:05:17', 'False', 'MOBILE');

INSERT INTO DYNAMIC VALUES('19232', '879126',

'ABSENT', 'STRAIT OF ISTANBUL', '2019-03-09 03:14:02', '5', 'West', 'True',

'29.35', '41.22', 'GEORGIA', '2021-05-01 02:06:18', 'False', 'STATIC');

7. Data Dictionary

Name	Description	Role	Type	Format	Nulls ?	Default Value
MMSI	A unique identifier for each ship	E - Ship, A - Primary Key, R - Transmits 0:M	float	1 to 9 digits	Yes	185739193 56
name	What the ship is referred by	E - Ship, A - Common Identifier	String	words	Yes	Queen Mary
length	the length of the ship	E - Ship, A - one factor of the ships full size	float	in meters	Yes	230
width	the width of the ship	E - Ship, A - one factor of the ships full size	float	in meters	Yes	80
imo	international maritime organization code	E - Ship, A - identifier	binary	N/a	Yes	294788421
device accuracy	the accuracy of info from the ship	E - Dynamic, A - denotes likelihood of miscommunication	decimal	<= 1	Yes	"True"
device type	the device the ship is equipped with	E - Dynamic, A - the type of device	string	N/a	Yes	Static
timestamp	the time the information was sent	E - Dynamic, A - time info was sent, R - Transmits 1:N	timestamp	N/a	Yes	"2021-04-07 04:10:05"
speed	the speed the boat is moving	E - Dynamic, A - ship's current speed, R - Traffic 0:M	float	in Knots	Yes	12
direction	the direction of the boat	E - Dynamic, A - the ship's direction, R - Traffic 0:M	String	degrees N, S, W, E	Yes	East
status	what the ship is currently doing	E - Dynamic, A - ship's status, R - Traffic 0:M, Dock's 0:N	String	num 0-15 with status	Yes	"True"

max_longitude	longitude of the ship location	E - Dynamic, A - longitude of the ship	float	in degrees	Yes	29.35
min_latitude	latitude of the ship location	E - Dynamic, A - latitude of the ship	float	in degrees	Yes	11.17
destination	where the ship is heading	E -Dynamic, A - ships destination, R - Dock's 0:N	String	N/a	Yes	Georgia
eta	the time the ship will arrive at its destination	E - Dynamic, A - estimated time of arrival, R - Dock's 0:N	datetime	N/a	Yes	"2021-03-11 11:14:05"
name	Passage Name or type	E - Waterway, A - name/type	String	N/a	Yes	Strait of Istanbul
capacity	size of the passage way	E - Waterway, A - capacity, R - Traffic 1:1	String	in square meters	Yes	30000
max_longitude	longitude of the port location	E - Waterway, A - max_longitude	String	in degrees	Yes	22.37
max_latitude	latitude of the port location	E - Waterway, A - max_latitude	String	in degrees	Yes	13.45
min_longitude	longitude of the port location	E - Waterway, A - min_longitude	String	in degrees	Yes	14.17
min_latitude	latitude of the port location	E - Waterway, A - min_latitude	String	in degrees	Yes	18.19
max_longitude	longitude of the port location	E - Port, A - max_longitude	String	in degrees	Yes	41.43
max_latitude	latitude of the port location	E - Port, A - max_latitude	String	in degrees	Yes	12.13
min_longitude	longitude of the port location	E - Port, A - min_longitude	String	in degrees	Yes	32.34
min_latitude	latitude of the port location	E - Port, A - min_latitude	String	in degrees	Yes	17.65
name	the name of the port	E - Port, A - Name	String	N/a	Yes	"Port of Antwerp"
capacity	how many ships can dock at a time	E - Port, A - Ship Capacity, R - Dock's 0:1	float	in feet	Yes	40

8. SQL

1. Lists the current location, estimated time arrival, and destination of a ship with the MMSI of 11873

Select LONGITUDE, LATITUDE, ETA, DESTINATION

From DYNAMIC

Where SHIP_MMSI = 11873;

```
+-----------+----------+---------------------+-------------+
| LONGITUDE | LATITUDE | ETA                 | DESTINATION |
+-----------+----------+---------------------+-------------+
| 4.40      | 51.21    | 2021-05-01 03:09:16 | SHANGAI     |
+-----------+----------+---------------------+-------------+
1 row in set (0.0004 sec)
```

2. Lists the location and ship capacity of the Port of Singapore

Select MAX_LONGITUDE,MAX_LATITUDE,MIN_LONGITUDE,MIN_LATITUDE, CAPACITY

From PORT

Where NAME ="Port Of Singapore"

```
+---------------+--------------+---------------+--------------+----------+
| MAX_LONGITUDE | MAX_LATITUDE | MIN_LONGITUDE | MIN_LATITUDE | CAPACITY |
+---------------+--------------+---------------+--------------+----------+
| 103.58        | 1.13         | 103.20        | 1.03         |       45 |
+---------------+--------------+---------------+--------------+----------+
1 row in set (0.0005 sec)
```

3. Lists the type of ships and their organizational codes that are headed to Georgia

Select s.NAME, s.SHIP_TYPE, s.IMO

From SHIP as s, DYNAMIC as d

Where s.MMSI=d.SHIP_MMSI and d.DESTINATION = "Georgia";

```
+------------+-----------+----------+
| NAME       | SHIP_TYPE | IMO      |
+------------+-----------+----------+
| QUEEN MARY | TANKER    |  2342360 |
| CUMHURIYET | PRIVATE   | 19231900 |
+------------+-----------+----------+
2 rows in set (0.0004 sec)
```

4. Lists the ship type, speed, and the destination of the ship called Evergiven

Select s.SHIP_TYPE, d.SPEED, d.DESTINATION

From SHIP as s, DYNAMIC as d

Where s.MMSI=d.SHIP_MMSI and s.NAME= "Evergiven";

```
+------------+-------+-------------+
| SHIP_TYPE  | SPEED | DESTINATION |
+------------+-------+-------------+
| COMMERCIAL | 5     | SHANGAI     |
+------------+-------+-------------+
1 row in set (0.0004 sec)
```

5. Lists the name of the ports with a ship capacity greater than 15

Select NAME, CAPACITY

From PORT

Where CAPACITY > 15;

```
+--------------------+----------+
| NAME               | CAPACITY |
+--------------------+----------+
| PORT OF ANTWERP    |       20 |
| PORT OF SINGAPORE  |       45 |
+--------------------+----------+
2 rows in set (0.0004 sec)
```

6. Lists all ships and their navigational status, ETA, and device type

Select s.NAME, d.STATUS, d.ETA, d.DEVICE_TYPE

From SHIP as s, DYNAMIC as d

Where s.MMSI = d.SHIP_MMSI;

```
+------------+--------+---------------------+-------------+
| NAME       | STATUS | ETA                 | DEVICE_TYPE |
+------------+--------+---------------------+-------------+
| EVERGIVEN  | True   | 2021-05-01 03:09:16 | MOBILE      |
| QUEEN MARY | True   | 2021-05-01 04:05:17 | MOBILE      |
| CUMHURIYET | True   | 2021-05-01 02:06:18 | STATIC      |
+------------+--------+---------------------+-------------+
3 rows in set (0.0004 sec)
```

7. Lists the name of the ships whose length is less than 300m in descending order

Select NAME, LENGTH

From SHIP

Where Length < 300 ORDER BY LENGTH DESC;

```
+-----------------+--------+
| NAME            | LENGTH |
+-----------------+--------+
| EVERGIVEN       |    280 |
| QUEEN ELIZABETH |    190 |
| CUMHURIYET      |     65 |
+-----------------+--------+
3 rows in set (0.0004 sec)
```

8. Lists all the ships and their MMSI that are Commercial type

Select NAME, MMSI

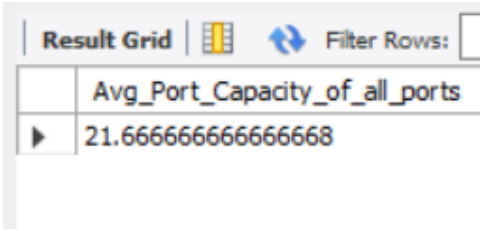
From SHIP

Where SHIP_TYPE="Commercial";

```
+------------------+-------+
| NAME             | MMSI  |
+------------------+-------+
| EVERGIVEN        | 11873 |
| ECLIPSE          | 18462 |
| EMMA MAERSK      | 49068 |
| OASIS OF THE SEAS | 56924 |
+------------------+-------+
4 rows in set (0.0004 sec)
```

9.Find the average ship capacity for all the ports

select avg(capacity) as 'Avg_Port_Capacity_of_all_ports' from port;



10. Calculate the average length and width for each type of ship

select ship_type, avg(length) as average_length, avg(width) as average_width from ship group by ship_type;

ship_type	average_length	average_width
COMMERCIAL	331.25	135
TANKER	270	95
PRIVATE	65	14

11.Find out average Speed per each ship type In each direction of Travel.

select d.direction, s.ship_type, avg(d.speed) from dynamic d join ship s on d.ship_mmsi= s.mmsi group by d.direction, s.ship_type;

direction	ship_type	avg(d.speed)
East	COMMERCIAL	5
West	TANKER	5
West	PRIVATE	5

12. Identify the busiest passage Latitude and Longitude.

select longitude,latitude,count(ship_mmsi) from dynamic group by longitude, latitude;

longitude	latitude	count(ship_mmsi)
4.40	51.21	1
11.26	57.53	1
29.35	41.22	1

13. Identify the busiest ports based on Incoming Ships.

select destination,count(ship_mmsi) as ship_count from dynamic group by destination order by ship_count desc;

destination	ship_count
GEORGIA	2
SHANGAI	1

14. Mean Delay between ETA and original arrival time per each port.

select destination, avg(timestampdiff(HOUR,TIME_STAMP,ETA)) as average_diiference_in_ETA_and_arrival from dynamic group by destination;

destination	average_diiference_in_ETA_and_arrival
SHANGAI	574.0000
GEORGIA	14820.0000

15. Display waterway names and how many more ships can safely pass through each one.

select waterway_name, capacity - count(ship_mmsi) as current_capacity from dynamic, waterway where waterway_name = name and name not like 'absent' group by waterway_name;

waterway_name	current_capacity
KATTEGAT STRAIT	199
STRAIT OF ISTANBUL	44

16. Display the last known locations of ship's not at a port.

select ship_mmsi, longitude, latitude from dynamic where port_name like 'absent';

	ship_mmsi	longitude	latitude
▶	16930	11.26	57.53
	19232	29.35	41.22

17. list the name and average space (longitude x latitude) allocated to each ship in the water way.

select name as waterway, ((max_latitude-min_latitude)*(max_longitude-min_longitude))/capacity as avg_space_occupied_by_ship from waterway where name not like 'absent';

	waterway	avg_space_occupied_by_ship
▶	KATTEGAT STRAIT	0.02841700000000001
	STRAIT OF ISTANBUL	0.06727111111111113

9. Time Log

Task	Hours Spent	Name
Conceptual Basis for System Requirements	1	Furkan
System/Client Requirements, Draft	2.5	Furkan
Contextual Data Flow Diagram, Draft	1	Furkan
ER/EER diagram	3	Mahdsiah
Normalized DB Model; including updated normalizations	3.5	Mahdsiah
recommendation on DBMS and implementation ready DB - Model	2.5	Bhargav
SQL Queries; 1-8	3.5	Victor
Data Dictionary	3.5	Mahdsiah
Implementation Ready DB Model Updates; Data Insertion	2.5	Furkan
ER Diagram Update	1.5	Furkan
DBMS recommendation - updates	1.5	Bhargav
SQL Queries; 1-8 - update with new tables	1.5	Victor
Implementation Ready DB Model Updates; updated foreign keys	.25	Mahdsiah
Additional SQL Queries; 9-14	1.5	Bhargav
Additional SQL Queries; 15-17	1	Mahdsiah
Data Dictionary Update, Final Editing	1.5	Furkan

M. Furkan Oruc Approved
Bhargav Nelluri Approved
Mahdsiah Khaalik Approved
Victor Lam Approved

Appendix

Feedback for Project Phase-0

Submission Feedback

Overall Feedback

Hi,
Please proceed with Idea; 1: **Maritime Traffic Database**
Follow guidelines given by professor for Phase-1.