Department of Computer Science and Engineering

CSE 2211: Database Management System - 1 Lab

Project Title:Sea-Port Management

Submitted to:

Dr. Md. Mustafizur Rahman
Dr. Muhammad Ibrahim

Submitted by:

Md.Sakib Ur Rahman
Roll:37

Contents:

A.Project Title: 3
B.Project Area Major: 3
C.Project Area Minor: 3
D.Brief Description of the Database: 3
E. Detailed Description of the Database: 4
F.Expected Query Outputs from the Project: 7
G. List of Tables with schema (example, Dept = (dept_id, dept_name, dept_budget)): 8
H… E-R Diagram: 9
I. List of Functional Dependencies to be maintained in the project: 11
J..Table schema 12
K.Table-level and project level expected constraints : 24
L. SQLs to implement the project with example outputs: 25
M. Finding the Normal Form (1st/2nd/3rd/3+ or BC-NF) 25
N. Future Works and Conclusions 27

A.Project Title:

Sea Port Management System

B.Project Area Major:

Transportation ,Maritime Logistics and Operations,Exportation.

C.Project Area Minor:

Port Management, Cargo Handling, Sailing Operations,Container Handling.

D.Brief Description of the Database:

The Sea Port Management System is a sophisticated and integrated database solution meticulously crafted to enhance and streamline operations within the dynamic seaport environment. Actually it has many parts.From the export company to import company.But Here ,I only perform the Export System.That means you can perform only export from our port.This comprehensive database is strategically designed to efficiently manage diverse facets of port activities, encompassing container handling, sailing operations, cargo tracking, and the allocation and utilization of crucial port resources. The system aims to provide a centralized platform for real-time monitoring, analysis, and decision-making, fostering heightened operational efficiency and resource optimization within the maritime logistics domain.

E. Detailed Description of the Database:

The Export part of the Sea Port Management System is a robust and integrated database designed to facilitate efficient operations within a seaport

environment. This comprehensive system encompasses various tables, each serving a specific purpose in managing and organizing essential data related to port activities.Let's have a short look at what problem it will solve for you.
Let's your friend need to send a supercomputer from Bangladesh to Austrailia.The he first need to go any transport company.They will receive full destination location and full information of his goods.Then the transport company sends the goods to a export /import company .The company will receive many goods like this .Then they will merge goods of same destination.Then they will call a Freight forwarding company to send a container.From now,we build our database.The company will give a container from a ship company to the export company.Then the container with full of goods will come to the close port .Here we choose the Chittagong port .The cargo truck with the container will enter the port if the port accept it by verifying license,goods status that means if the containers contain how type of goods .It can be explosive,normal,food ,oil,gas or dangerous chemical.Then will accept the container by following the rule of the country.Then the cargo truck go to the yard for unloading the container .This unloading will be proform.After choosing a sailing where the information of a ship that is ready for sailing are kept.The freight forward company choose a destination port for the import address.Then the container are loaded by the harbor pilots to the ship.
Let's see what will be performed by our database .
Before starting ,one important thing is that we will work for our CHITTAGONG PORT as a sea port.

- Manages all the worldwide export-import company information that are dealing with our port.
- Manages the information of all the ports that are close to an export-import company.
- Contains details about different ports.
- Contains information about harbor pilots assisting with ship navigation.
- Manages all the worldwide shipping companies that can provide ships and containers to our port.

- Stores data about crane operators working in the port.
- Stores the information of the cargo truck including its driver that carries the container from the export office to port.
- Stores the full container information that comes to our port.
- Manages all information how the container is unloaded from the truck to where and who will perform and when.
- Manages freight forwarding information, including cargo details, drivers, and loading times.
- It also has extra management for the freight forwarder company.That is they can send the container not only to our port but also all the port.
- Tracks the unloading of containers in the yard.
- We can find all the containers that are coming for export with all information such as which company's container ,when,which export-import company,which truck carries it,which driver drives it,when it is unloaded from the truck,which crane driver performs it and where it is kept.
- Monitors the total yard capacity for incoming containers.
- Manages all the ships of many companies that have deal id for this port.
- Contains all the sailing information such as which ship performed it,where it will go,where it was anchored,how many containers it contained,how total weight it carried,which container it carried,when it left the port,which captain drove the ship.
- Manages details about ships and their capacities.
- Stores the information that how many containers are already loaded to ship,how many containers are yet to load,which harbor pilots load the container from the yard to ship,when.
- Tracks the status of anchorages within the port.
- Contains all the anchorage information when it was used ,which sailing,which ship,when it became empty.how many ships used it.
- Contains information about ship departures and their destinations.

- Contains information about harbor pilots assisting with ship navigation.
- Manages cargo loading information, including harbor pilots and loading times.
- You can find all the deal that are done by a company.
- Manages all the sailing that successfully leaves the port.For this ,we can find how many containers are left from the port including time.
- Tracks leaves taken by ships, affecting their availability.
- Actually you can find all the real world information about the work that is done for exporting a container.
- All the tasks that are done in every event ,are kept in their time.

Let’s see which cannot work by our port.

- In a real world application,a port contains all the information from accepting the goods from a person to sending the goods to the specific person.But here we only perform the event from the export company to the leaving of a ship for a specific destination port.
- You cannot find the information about how the goods inside of the container are processed.
- You cannot find any information about the goods inside the container.Just you can find what is its type.If is harmful or not .
- You cannot even find the information about who is the real owner of the goods inside the container.
- You cannot find the information about the police who check the container and also who enter the container to the port.
- Also I have not added any expensive cost system for the container.

F.Expected Query Outputs from the Project:

- Total Neet Weight Handled by Each Company

```
SELECT company_id, SUM(neet_weight) AS total_neet_weight
FROM ff
GROUP BY company_id;
```

- Calculate the average departure time for all sailings in the system.

```
 SELECT AVG(dept_time) AS average_departure_time
FROM sailing;
```

- List of Ships and Their Departure Status

```
SELECT ship.ship_id, ship.ship_name, sailing.dept_time
FROM ship
LEFT JOIN sailing ON ship.ship_id = sailing.ship_id;
```

- Retrieve details of containers and their unloading information, including containers without unloading records.

```
SELECT container.id, container.sizee, unload_incoming_yard.u_time
FROM container
LEFT JOIN unload_incoming_yard ON container.id =
unload_incoming_yard.f_id;
```

- Retrieve all records from the `ship_company` table.

```
SELECT * FROM ship_company;
```

- Retrieve the names of all ships along with their total capacity weight.

```
SELECT ship_name, capacity_weight FROM ship;
```

- Find the average neet_weight of containers for each company from the `ff` table.

```
SELECT company_id, AVG(neet_weight) AS avg_neet_weight
FROM ff
GROUP BY company_id;
```

- List all containers along with the driver names and loading times from the `ff` table.

```
SELECT f.*, d.name AS driver_name
FROM ff f
JOIN crane_drivers d ON f.driver_id = d.id;
```

- Find the total size available in each incoming yard.

```
SELECT yard_id, SUM(total) AS total_size
FROM incoming_yard
GROUP BY yard_id;
```

G. List of Tables with schema (example, Dept = (dept_id, dept_name, dept_budget)):

Table no.1
exim(id,location,name);
Table no.2
ship_company(id,name,location);
Table no.3
container(id,company_id,active_status,sizee);
Table no.4
port(id,name,location,area);
Table no.5
ff(f_id,container_id,company_id,exp_id,imp_id,port_id,element ,neet_weight,driver_id,truck_id,loading_time,arrival_time);
Table no.6
incoming_yard(id,total);
Table no.7
crane_drivers(id,name);
Table no.8
harbor_pilots(id,name);
Table no.9
unload_incoming_yard(f_id,yard_id,crane_driver_id,u_time);
Table no.10
ship(ship_id,company_id,ship_name,capacity_size,capacity_weight,status);
Table no.11
anchorage(id,status);
Table no.12

sailing(s_id,ship_id,company_id,anchorage_id,to_location,dept_time, status);

Table no.13

exim_port(exim_id,port_id);

Table no.14

load(sailing_id,f_id,harbor_pilots_id,l_time);

Table no.15

leave(sailing_id,l_time);

H… E-R Diagram:

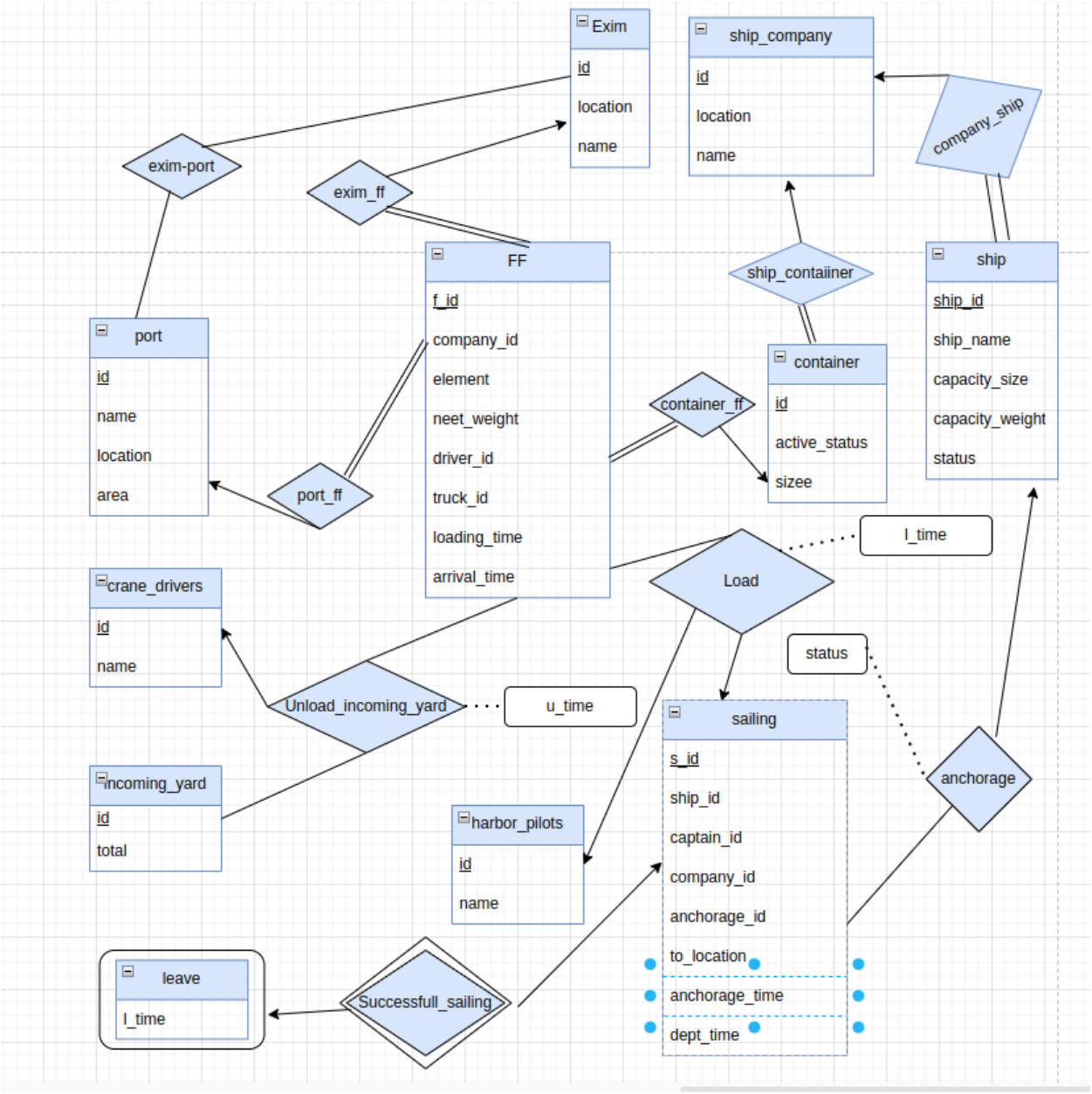


Fig:E-R diagram…

I. List of Functional Dependencies to be maintained in the project:

Let's see the functional dependencies in all tables;

Exim Table:

1.id ⟶ name,location.

Ship_company:

1.id ⟶ name,location.

Container:

1.id,company_id ⟶ active_status,sizee.

2.ship_company.company_id ⟶ company_id.

Port:

1.id ⟶ name,location,area.

FF.

1.f_id,port_id ⟶ container_id,company_id,exp_id,imp_id,port_id,element,neet_weight,driver_id,truck_id,loading_time,arrival_time.

2.container.id,container.company_id ⟶ container_id,company_id.

3.exim.exp_id ⟶ exp_id

4.exim.imp_id ⟶ imp_id

5.port.id ⟶ port_id

Incoming_yard:

1.id ⟶ total.

Crane_drivers:

1.id ⟶ name

Harbor_pilots:

1.id ⟶ name

Unload_incoming_yard:

1.f_id ⟶ yard_id,crane_driver_id,u_time

2.incoming_yard.id ⟶ yard_id

Ship:

1.ship_id,company_id ⟶ ship_name,capacity_size,capacity_weight,status

2.ship_company.id ⟶ company_id

Anchorage:

1.id ⟶ status

Sailing:

1.s_id → ship_id,company_id,captain_id,anchorage_id,to_location,anchorage_time,dept_time,status.

2.ship.ship_id,ship.company_id → ship_id,company_id

3.anchorage.id → anchorage_id

Exim_port:

1.exim_id,port_id → .

2.exim.id → exim_id

3.port_id → port_id

Load:

1.sailing_id,f_id → harbor_pilots_id,l_time

2.sailing.s_id → sailing_id

3.harbor_pilots.id → harbor_pilots_id

Leave:

1.sailing_id → l_time

2.sailing.s_id → sailing_id

J..Table schema

table.1:Exim

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(255)	Primary key	Contains both the import and export id
2	location	varchar2(255)		
3	name	varchar2(255)		

Purpose: We perform export or import through an Export or Import company.This table stores their information.
Type: The id will be a global information.The location and name are corresponding to that id.We can find the name and location from the id.
Ex:VALUES ('1001', 'BANGLADESH', 'CSEDU');

Table..2: ship_company

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(255)	Primary key	Contains both the import and export id
2	name	varchar2(255)		
3	location	varchar2(255)		

Purpose: As we use the sea so we need ships,containers etc.So first we need ship company that will provide us these things.

Type: The id is a global unique information.It helps us to know this company name and location.

Ex:VALUES ('1001', 'EVER GREEN', 'TAIWAN');

Table..3: container

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(255)	Primary key	Contains a container id.
2	company _id	varchar2(255)	Primary key,foreign key references ship_company(id)	Contains the company_id that has this container.
3	active_status	varchar2(255)		Here we can see if the container is Active,service or inactive.Active means we can use it,service means it is now using,and inactive means it needs servicing.

4	sizee	number		It shows the size of the container which we need at the time of cost ,space on the ship or yard etc. In sq-m.

Purpose: This table helps us to identify each container through which we contain our goods.
Type: It will store a unique (id+company_id) .Active status helps us to use this container.And size will help us to maintain cost.
Ex:VALUES ('1003', '1002', 'ACTIVE', 300);

Table..4: port

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(255)	Primary key	Contains a globally unique port address.
2	name	varchar2(255)		Port name
3	location	varchar2(255)		
4	area	number		Total area of that port in sq-m.

Purpose : This table provides us useful information about a port.
Type:It contains an address of the port where the container will come for ship along with name ,location and area in sq-m.
Ex:values ('2003','Chittagong','BanglaDesh','11000');

Table..5:ff

S/N	Attribute	Data Type	Constraint	Comments
1	f_id	varchar2(50)	constraint freight_id primary key(f_id,port_id)	Contains a globally unique freight_forward id along with a port_id.

2	container_id	varchar2(50)	CONSTRAINT fk_container FOREIGN KEY (container_id ,company_id) REFERENCES container(id, company_id) ,	Container specific address along with a company address.
3	company_id	varchar2(50)	DO	do
4	exp_id	varchar2(50)	CONSTRAINT fk_exim_exp FOREIGN KEY (exp_id) REFERENCES exim(id)	From which the container will export.
5	imp_id	varchar2(50)	DO	To where the container will import.
6	port_id	varchar2(50)	Primary key, CONSTRAINT fk_port FOREIGN KEY (port_id) REFERENCES port(id)	A port address where the container will go for exporting.
7	element	varchar2(50)		Specifies the types of the good which the container carry.types can be explosive,normal.It is used to indicate how It should be carried,managed.

8	neet_weight	number		Total weight of that container with the goods.
9	driver_id	varchar2(50)		Contains the truck driver license or address who carry the container from the export office to port.
10	truck_id	varchar2(50)		Specifies the truck_id on which the container is carried.
11	loading_time	TIMESTAMP		Contains the time when the container is filled with goods and is about to leave from the export office.
12	arrival_time	TIMESTAMP		The expected time when the container should reach on destination port or import office.

Purpose: This table is used when a container with its goods is about to leave an export or import office to be exported.This table is used by a FREIGHT FORWARDer company to choose a container from a ship company.It contains every information that is needed to export a container. We can find all the freight forward id from a port.
Type: We can use a container from a company which is Active.This is done by a trigger function named ff_insert_trigger and a procedure to update status to Service after the selection of a container named update_container_status.
EX:(f_id, container_id, company_id, exp_id, imp_id, port_id, element,neet_weight,driver_id,truck_id,loading_time, arrival_time) VALUES ('1003', '1002', '1002', '1001', '1002', '3003', 'Normal',5000,'100001','200001',CURRENT_TIMESTAMP, TIMESTAMP '2024-12-12 12:30:00');

From now we will consider that port id 2003 ‘chittagong port’ is our port and every material belongs to our port.

Table..6: incoming_yard

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(3)	Primary key	This is the yard address of our Chittagong port.
2	total	number		Total size .It helps us to know how many containers it can contain.

Purpose: After the truck with a container reached a port.[our chittagong port] ,it first goes to the incoming_yard to unload the container.The container will be here until it finds a ship that will carry it to its destination. We can check how many remaining spaces in the yard to contain the container.
Type: It contains the total remaining size according to the yard id.
Ex:(id, total) VALUES ('A', 80000);

Table. 7: crane_drivers

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(20)	Primary key	This is the unique driver id.
2	name	varchar2(100)		This is the driver's name.

Purpose: This table helps us to find all the crane driver information in our port.The crane drivers are the people who unload or load the container from the cargo truck.
Type:id is a license and the name is the valid name of the driver.
Ex: values('1001','Mr ab');

Table..8: harbor_pilots

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(20)	Primary key	This is the unique driver id.
2	name	varchar2(100)		This is the driver's name.

Purpose. Actually every harbor system has a group of people .But I only show the leader of the group and recognize him as a harbor pilots.This harbor is used to upload ,unload containers to /from ships.
Type: The leader's unique address who is responsible for taking the container from the yard to the ship.And the name belongs to his group name.
Ex.insert into harbor_pilots values('2001','faith');

Table..9: unload_incoming_yard

S/N	Attribute	Data Type	Constraint	Comments
1	f_id	varchar2(50)	Primary key	Unique f_id .And we already know that this f_id belongs to our port .
2	yard_id	varchar2(3)	CONSTRAINT fk_ex FOREIGN KEY (yard_id) REFERENCES incoming_yard(id),	Place address where the containers are kept for loading to ship.
3	crane_driver_id	varchar2(20)		This is the crane driver's id who is responsible for unloading that container.
4	u_time	TIMESTAMP		This is the time when the container was unloading from the truck.

Purpose: This table will help us if we found anything wrong with the container.Then we can know who was responsible for unloading the container.Also we find information about when the container was reached to our port and when unloading.We also find where a container was unloading.This will help us to find the container at the time of loading for the ship.The main purpose for creating this table is that after the container with a f_id come to the port,then the manager of the export company ask the

port manager for a ship that will carry the container for the import companies reachable port.And here the container waiting for ship.

Type: Every yard has a limited space.If the container size is less than the remaining yard space then it can be unloaded.Also we have to update the yard space after giving space for the container.This is done by a trigger function named insert_unload_incoming_yard.Also a crane drive can only unload a container within two hours.
Ex:values('1001','A','1001',CURRENT_TIMESTAMP);

Table..10: ship

S/N	Attribute	Data Type	Constraint	Comments
1	ship_id	varchar2(50)	Primary key	Contains the address or id of a ship.
2	company_id	varchar2(50)	Primary key, foreign key(company_id) references ship_company(id)	Contains the company id.
3	ship_name	varchar2(50)		Ship name according to the id.
4	capacity_size	number		The total capacity size that the ship can carry.The space where the containers will be kept.
5	capacity_weight	number		The total weight of the container that the ship can carry.
6	status	varchar2(20)		Current status of the ship.If the status is Active ,then we can use them.

Purpose: For sailing,we need a ship that will carry the container .This table stores all the information about the ship that we will use.

Type: We can use a ship if its status is Active.
Ex: values('1001','1001','pirates of caribbean',50000,80000,'ACTIVE');

Table...11: anchorage

S/N	Attribute	Data Type	Constraint	Comments
1	id	varchar2(3)	Primary key	This is the unique terminal location address of our port.
2	status	varchar2(20)		The current status of this termina.Is it empty or filled with ships.

Purpose: This table helps us to choose a terminal for anchoring a ship.By sitting here,a ship is filled with containers .If the terminal is empty then we can use it .When there is already a ship anchored ,then the terminal status is filled and we cannot use it ,until the ship is leaved .
Type: It stores a unique id and status.
Ex:values('a1','empty');

Table…12: sailing

S/N	Attribute	Data Type	Constraint	Comments
1	s_id	varchar2(50)	Primary key	Contains the address of a unique sailing..
2	ship_id	varchar2(50)	FOREIGN KEY (ship_id, company_id) REFERENCES ship(ship_id, company_id),	Contains the ship id.
3	company_i	varchar2(50)	DO	Contains the company id.

	d			
4	captain_id	varchar2(50)		The group of captains or drivers who will drive the ship.
5	anchorage_id	varchar2(3)	FOREIGN KEY (anchorage_id) REFERENCES anchorage(id)	The terminal location where the ship is waiting for loading containers.
6	to_location	varchar2(50)		Where the ship will go.
7	anchorage_time	TIMESTAMP		The Starting time when the ship comes for anchoring.
8	dept_time	TIMESTAMP		The time when the ship will leave the terminal or start its sailing.

Purpose: When a ship ,with a destinal ,with a departure time waiting in the terminal for loading with containers, is called sailing.This table shows us all the sailing that has happened to this port.We have to choose sailing to import the containers according to our destination location.
Type: This table contains a ship according to a ship company in a terminal from a time to dept_time.the dept_time can be changed.We cannot choose a ship that is not Active.We cannot also choose a terminal that is not Active.This is done by a trigger called check_and_update_status.

Ex:values('1001','1001','1001','1001','a1','2001',CURRENT_TIMESTAMP,TIMESTAMP '2024-12-12 12:30:00');

Table.13: exim_port

S/N	Attribute	Data Type	Constraint	Comments
1	exim_id	varchar2(3)	Primary key,foreign key(exim_id) references	The address of an export or import company.

			exim(id),	
2	port_id	varchar2(20)	Primary key,foreign key(port_id) references port(id),	The address of a port.

Purpose: This table helps us to find all the port_id which is according to an export or import company or the company who can deal with which port.It is an extra table.
Type:the exit_id comes from the Exim table and the port_id comes from the port table.
Ex: values('1001','2003');

Table ..14: load

S/N	Attribute	Data Type	Constraint	Comments
1	sailing_id	varchar2(50)	Primary key,FOREIGN KEY (sailing_id) REFERENCES sailing(s_id),	Contains the address of a unique sailing..
2	f_id	varchar2(50)	Primary key, FOREIGN KEY (f_id) REFERENCES ff(f_id),	Contains the f_id.
3	harbor_pilots_id	varchar2(50)	FOREIGN KEY (harbor_pilots_id) REFERENCES harbor_pilots(id)	Contains the group of harbor pilots that are responsible to upload the container from ground to ship.
4	l_time	TIMESTAMP		The loading time ..Actually it is the

				time when the container is uploaded to the ship.

Purpose:This table helps us to upload a container with f_id to the ship that will reach its destination port.
Type:Here we use a trigger function that a harbor_pilots can only upload a container within every two hours.the l_time must be less than the sailing.Also when the ship weight_capacity is filled ,then we cannot load any container.Also if the sailing is available ,then we can only use the sailing.
Ex:values('1006','1002','2003',CURRENT_TIMESTAMP);

Table..15:leave

S/N	Attribute	Data Type	Constraint	Comments
1	sailing_id	varchar2(50)	Primary key,foreign key(sailing_id) references sailing(s_id)	Contains the address of a unique sailing..from the sailing table.
2	l_time	Timestamp		Contains the successful leaving time of the ship.

Purpose: This table helps us to find the successful leave of a sailing.When the sailing is complete or the ship is leaving ,then we cannot use the ship or sailing.
Type: Here we use a trigger so that when we insert a sailing_id in that table that means we cannot reuse the sailing for taking containers.The trigger name is after_leave_insert.Then we also update the anchorage_id or the terminal id to empty where the sailing was anchored.because the anchor is now fully free.
Ex:values('1002',CURRENT_TIMESTAMP);

K.Table-level and project level expected constraints :

I already show the primary and foreign key constraint on the table schema. Here I will show other constraints .

- To use a container we first need a ship company that will give us a container.But to use the container ,the container must be Active.Active means we can use it,Inactive means it needs repairing and Service means it is now being used.

- To export something we need an export-import company that will collect goods to import.So we need a valid export-import company.
- We need a port from where the import company receives its container.
- At the time of making freight forwarding id..we have a constraint that the according container must be active and after making the f_id ,the status will be service. It is done by a trigger and procedure function named ff_insert_trigger and update_container_status.
- The container can be unloaded in the incoming yard by a valid crane_driver.Also a crane driver can be used within two hour in once.Also the remaining space of the yard must be greater than the container space.These is done by a trigger function name insert_unload_incoming_yard.
- For sailing ,we need an active ship and we need to anchor it in an empty anchor.And after successfully anchored,we modify the status of the anchorage to fill and make the ship status to service.These are done by a trigger function name check_and_update_status.
- We use the sailing for exporting the container.So,we load our container according to our needed port.This is done by the load table.We can load a container to a sailing ship if the sailing status is available.Also a harbor pilot can only be used once within two hours. The loading time must be smaller than the departure time of the ship.Also a ship needs enough remaining space and weight to load the container.If the size or weight of the container is greater than the

remaining space or weight then we cannot load .These are done by a trigger function name check_load_time.

- Our last work is the leave table.Here we include the successful sailing using their time.It happens in any port that a ship leaves the port after or before its fixed time due to an unavoidable circumstances.When a ship is left ,then we have to modify the sailing status to unavailable so that no one can load any container to it .Also we have to make the using anchorage status to empty so that new sailing can use it.

L. SQLs to implement the project with example outputs:

exim table:1

```
CREATE TABLE exim (
    id VARCHAR2(255) PRIMARY KEY,
    location VARCHAR2(255),
    name VARCHAR2(255)
);
```

Output:

	ID	LOCATION	NAME
1	1001	BANGLADESH	CSEDU
2	1002	SINGAPORE	FERROCADIA
3	1003	India	Abey sle
4	1004	New Zealand	newz
5	1005	Pakistan	pak
6	1006	Austrailia	hoito

ship_company table:2

```
CREATE TABLE ship_company (
    id VARCHAR2(255) PRIMARY KEY,
    name VARCHAR2(255),
    location VARCHAR2(255)
```

);

Output:

	ID	NAME	LOCATION
1	1001	EVER GREEN	TAIWAN
2	1002	MAERSK	DENMARK
3	1003	YANG MING	CHINA
4	1004	SOTABDI...	CSEDU

container table:3

```
CREATE TABLE container (
    id VARCHAR2(255),
    company_id VARCHAR2(255),
    active_status VARCHAR2(255),
    sizee NUMBER,
    PRIMARY KEY (id, company_id),
    FOREIGN KEY (company_id) REFERENCES ship_company(id)
);
```

Output:

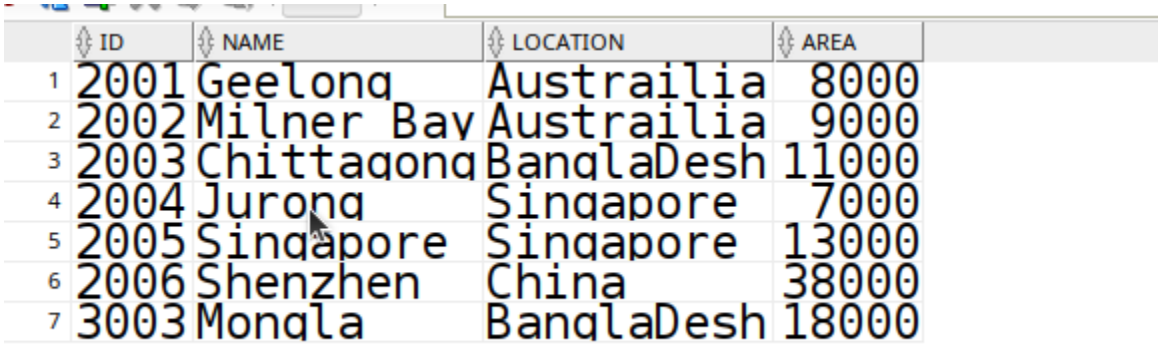
	ID	COMPANY_ID	ACTIVE_STATUS	SIZEE
1	1001	1001	SERVICE	100
2	1001	1002	INACTIVE	200
3	1002	1002	SERVICE	300
4	1003	1002	SERVICE	300
5	1002	1001	SERVICE	300
6	1003	1001	SERVICE	50
7	1004	1001	SERVICE	300
8	1005	1002	ACTIVE	200
9	1006	1002	ACTIVE	300
10	1007	1001	INACTIVE	100
11	1008	1002	ACTIVE	200
12	1009	1002	ACTIVE	300
13	1010	1001	SERVICE	100
14	1011	1001	SERVICE	200
15	1012	1001	SERVICE	300
16	1013	1001	ACTIVE	100
17	1014	1001	ACTIVE	200
18	1015	1001	ACTIVE	300
19	1010	1002	SERVICE	100
20	1011	1002	SERVICE	200
21	1012	1002	SERVICE	300
22	1013	1002	ACTIVE	100
23	1014	1002	ACTIVE	200
24	1015	1002	ACTIVE	300

port table:4

```
CREATE TABLE port (
    id VARCHAR2(255) PRIMARY KEY,
```

```
    name VARCHAR2(255),
    location VARCHAR2(255),
    area  number
);
```

Output:



	ID	NAME	LOCATION	AREA
1	2001	Geelong	Austrailia	8000
2	2002	Milner Bay	Austrailia	9000
3	2003	Chittagong	BanglaDesh	11000
4	2004	Jurong	Singapore	7000
5	2005	Singapore	Singapore	13000
6	2006	Shenzhen	China	38000
7	3003	Mongla	BanglaDesh	18000

freight forward table (ff):5

```
CREATE TABLE ff (
    f_id VARCHAR2(50) ,
    container_id VARCHAR2(50),
    company_id VARCHAR2(50),
    exp_id VARCHAR2(50),
    imp_id VARCHAR2(50),
    port_id VARCHAR2(50),
    element varchar2(50),
    neet_weight number,
    driver_id varchar2(50),
    truck_id varchar2(50),
    loading_time  TIMESTAMP,
    arrival_time TIMESTAMP,
        constraint freight_id primary key(f_id,port_id),

    CONSTRAINT fk_container FOREIGN KEY
(container_id,company_id) REFERENCES container(id,company_id),
```

```
   CONSTRAINT fk_exim_exp FOREIGN KEY (exp_id)
REFERENCES exim(id),
   CONSTRAINT fk_exim_imp FOREIGN KEY (imp_id)
REFERENCES exim(id),
   CONSTRAINT fk_port FOREIGN KEY (port_id) REFERENCES
port(id)
);
```

procedure.....

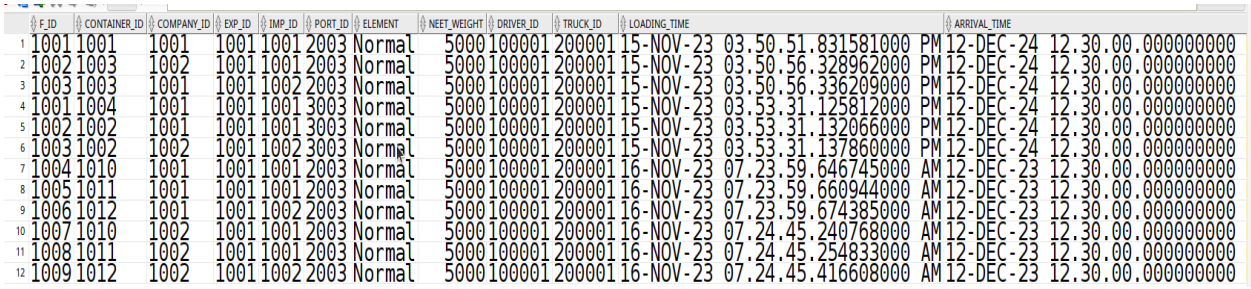
```
CREATE OR REPLACE PROCEDURE update_container_status(
        p_container_id varchar2,
        p_company_id varchar2
)
    as
    BEGIN
        UPDATE container
        set active_status='SERVICE'
        where id=p_container_id
        and company_id=p_company_id;


    Exception
        when no_data_found then
            raise_application_error(-20001,'container not
found');
        when others then
            raise_application_error(-20002,'an error
occured'||sqlerrm);
    End ;
```

trigger.....

```
CREATE OR REPLACE TRIGGER ff_insert_trigger
BEFORE INSERT ON ff
FOR EACH ROW
DECLARE
   v_active_status VARCHAR2(10);
BEGIN
   -- Check if the container is active
   SELECT active_status INTO v_active_status
   FROM container
   WHERE id = :NEW.container_id
     AND company_id = :NEW.company_id;

   -- If container is not active, raise an exception
   IF v_active_status <> 'ACTIVE' THEN
      RAISE_APPLICATION_ERROR(-20001, 'Container is not active.
Insert operation aborted.');
   END IF;

   -- Call the procedure to update container status
   update_container_status(:NEW.container_id, :NEW.company_id);
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      RAISE_APPLICATION_ERROR(-20002, 'Container not found.');
   WHEN OTHERS THEN
```

RAISE_APPLICATION_ERROR(-20003, 'An error occurred: ' ||
SQLERRM);
END ff_insert_trigger;

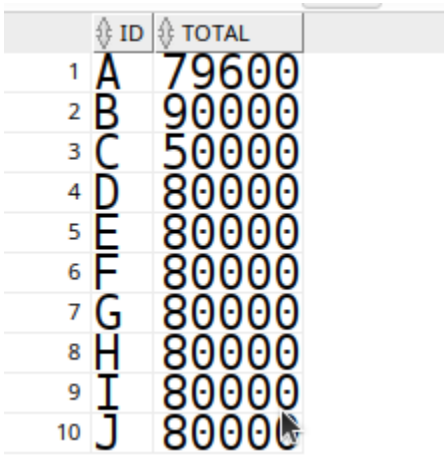
Output:



	F_ID	CONTAINER_ID	COMPANY_ID	EXP_ID	IMP_ID	PORT_ID	ELEMENT	NEET_WEIGHT	DRIVER_ID	TRUCK_ID	LOADING_TIME	ARRIVAL_TIME
1	1001	1001	1001	1001	1001	2003	Normal	5000	100001	200001	15-NOV-23 03.50.51.831581000 PM	12-DEC-24 12.30.00.000000000
2	1002	1003	1002	1001	1001	2003	Normal	5000	100001	200001	15-NOV-23 03.50.56.328962000 PM	12-DEC-24 12.30.00.000000000
3	1003	1003	1001	1001	1002	2003	Normal	5000	100001	200001	15-NOV-23 03.50.56.336209000 PM	12-DEC-24 12.30.00.000000000
4	1001	1004	1001	1001	1001	3003	Normal	5000	100001	200001	15-NOV-23 03.53.31.125812000 PM	12-DEC-24 12.30.00.000000000
5	1002	1002	1001	1001	1001	3003	Normal	5000	100001	200001	15-NOV-23 03.53.31.132066000 PM	12-DEC-24 12.30.00.000000000
6	1003	1002	1002	1001	1002	3003	Normal	5000	100001	200001	15-NOV-23 03.53.31.137860000 PM	12-DEC-24 12.30.00.000000000
7	1004	1010	1001	1001	1001	2003	Normal	5000	100001	200001	16-NOV-23 07.23.59.646745000 AM	12-DEC-23 12.30.00.000000000
8	1005	1011	1001	1001	1001	2003	Normal	5000	100001	200001	16-NOV-23 07.23.59.660944000 AM	12-DEC-23 12.30.00.000000000
9	1006	1012	1001	1001	1002	2003	Normal	5000	100001	200001	16-NOV-23 07.23.59.674385000 AM	12-DEC-23 12.30.00.000000000
10	1007	1010	1002	1001	1001	2003	Normal	5000	100001	200001	16-NOV-23 07.24.45.240768000 AM	12-DEC-23 12.30.00.000000000
11	1008	1011	1002	1001	1001	2003	Normal	5000	100001	200001	16-NOV-23 07.24.45.254833000 AM	12-DEC-23 12.30.00.000000000
12	1009	1012	1002	1001	1002	2003	Normal	5000	100001	200001	16-NOV-23 07.24.45.416608000 AM	12-DEC-23 12.30.00.000000000

incoming_Yard table:6.....

CREATE TABLE incoming_yard (
id VARCHAR(3) primary key,
total number
);
Output:



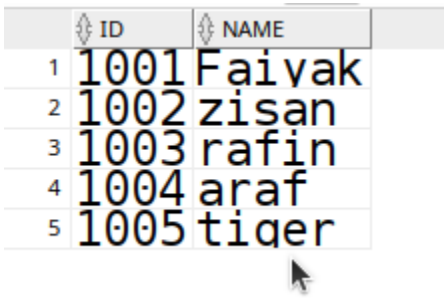
	ID	TOTAL
1	A	79600
2	B	90000
3	C	50000
4	D	80000
5	E	80000
6	F	80000
7	G	80000
8	H	80000
9	I	80000
10	J	80000

CraneDrivers table:7

CREATE TABLE crane_drivers (

```
    id VARCHAR2(20) PRIMARY KEY,
    name VARCHAR2(100)

);
```

Output:

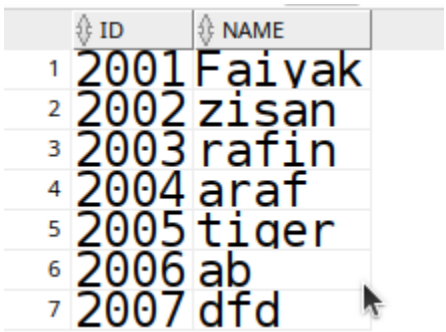


	ID	NAME
1	1001	Faiyak
2	1002	zisan
3	1003	rafin
4	1004	araf
5	1005	tiger

harbor_pilots table:8

```
CREATE TABLE harbor_pilots (
    id VARCHAR2(20) PRIMARY KEY,
    name VARCHAR2(100)

);
```

Output:



	ID	NAME
1	2001	Faiyak
2	2002	zisan
3	2003	rafin
4	2004	araf
5	2005	tiger
6	2006	ab
7	2007	dfd

unload_incoming_yard table:9

```
CREATE TABLE unload_incoming_yard (
    f_id VARCHAR2(50),
    yard_id VARCHAR2(3),
```

```
    crane_driver_id VARCHAR2(20),
    u_time TIMESTAMP,
    PRIMARY KEY (f_id),

        CONSTRAINT fk_ex FOREIGN KEY (yard_id) REFERENCES
incoming_yard(id)


);
```

trigger to unload container on incoming_yard.

```
CREATE OR REPLACE TRIGGER insert_unload_incoming_yard
BEFORE INSERT ON unload_incoming_yard
FOR EACH ROW
DECLARE
    v_container_id VARCHAR2(50);
    v_company_id VARCHAR2(50);
    v_container_size VARCHAR2(50);
    v_total_size NUMBER;
    v_last_u_time TIMESTAMP;
BEGIN
    -- Find container_id and company_id using f_id from ff table
    SELECT container_id, company_id
    INTO v_container_id, v_company_id
    FROM ff
    WHERE f_id = :NEW.f_id;

    SELECT sizee
    INTO v_container_size
    FROM container
    WHERE id = v_container_id
      AND company_id = v_company_id;
```

```
    -- Find total size using the yard id from the yard table
    SELECT total
    INTO v_total_size
    FROM incoming_yard
    WHERE id = :NEW.yard_id;

    -- Check if (total - sizee) is greater than or equal to 0
    IF (v_total_size - v_container_size) >= 0 THEN
        -- Check if there is a match in unload_incoming_container with
the new crane_driver_id
        SELECT MAX(u_time)
        INTO v_last_u_time
        FROM unload_incoming_yard
        WHERE crane_driver_id = :NEW.crane_driver_id;

        IF v_last_u_time IS NULL OR :NEW.u_time > v_last_u_time +
INTERVAL '2' HOUR THEN

            UPDATE incoming_yard
            SET total = v_total_size - v_container_size
            WHERE id = :NEW.yard_id;
        ELSE
            -- Raise an exception if the condition is not met
            RAISE_APPLICATION_ERROR(-20002, 'New u_time must
be greater than the last matched u_time plus 2 hours.');
        END IF;
    ELSE
        -- Raise an exception if the condition is not met
        RAISE_APPLICATION_ERROR(-20001, 'Insufficient yard space
for the container.');
    END IF;
END;
```

Output:

	F_ID	YARD_ID	CRANE_D...	U_TIME
1	1009	B	1006	19...
2	1007	A	1005	19...
3	1004	A	1002	19...
4	1005	A	1003	19...
5	1006	A	1004	19...

ship table:10

```
CREATE TABLE ship (
    ship_id VARCHAR2(50),
    company_id VARCHAR2(50),
    ship_name VARCHAR2(100),
    capacity_size NUMBER,
    capacity_weight NUMBER,
    status VARCHAR2(20),
    PRIMARY KEY (ship_id, company_id),
    foreign key(company_id) references ship_company(id)
);
```

Output:

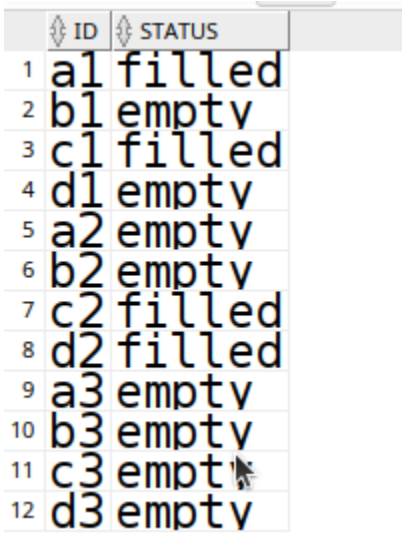
	SHIP_ID	COMPANY_ID	SHIP_NAME	CAPACITY_SIZE	CAPACITY_WEIGHT	STATUS
1	1001	1001	pirates of the caribbean	50000	80000	Service
2	1002	1001	CORAL	57000	82000	Service
3	1003	1001	shark bait	40000	71000	Service
4	1004	1001	the kraken	90000	88000	Service
5	1005	1001	hammerhead	10000	84000	INACTIVE
6	1006	1001	long weekend	20000	89000	ACTIVE
7	1002	1002	CORAL	57000	67000	Service
8	1003	1002	shark bait	40000	66000	ACTIVE
9	1003	1003	unsinkable	30000	82000	ACTIVE
10	1004	1003	the kraken	90000	88000	ACTIVE
11	1005	1004	hammerhead	10000	84000	INACTIVE
12	1006	1004	long weekend	20000	89000	ACTIVE

anchorage table:11

```
create table anchorage(
    id varchar(3) primary key,
```

```
    status  varchar(20)


);
```

Output:



	ID	STATUS
1	a1	filled
2	b1	empty
3	c1	filled
4	d1	empty
5	a2	empty
6	b2	empty
7	c2	filled
8	d2	filled
9	a3	empty
10	b3	empty
11	c3	empty
12	d3	empty

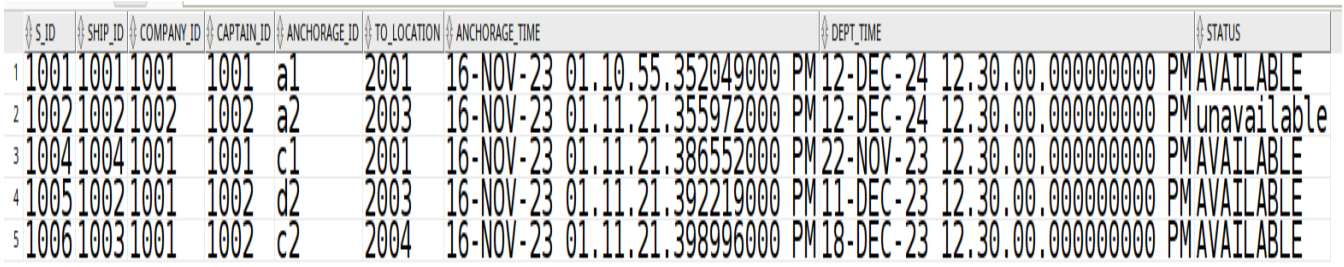
Sailing table:12

```
CREATE TABLE sailing (
    s_id VARCHAR2(50),
    ship_id VARCHAR2(50),
    company_id VARCHAR2(50),
    captain_id VARCHAR2(50),
    anchorage_id varchar(3),
    to_location VARCHAR2(100),
    anchorage_time TIMESTAMP,
    dept_time TIMESTAMP,
        status   VARCHAR2(50),
    PRIMARY KEY (s_id),
```

```
    FOREIGN KEY (ship_id, company_id) REFERENCES
ship(ship_id, company_id),
    FOREIGN KEY (anchorage_id) REFERENCES anchorage(id)

);

CREATE OR REPLACE TRIGGER check_and_update_status
BEFORE INSERT ON sailing
FOR EACH ROW
DECLARE
    v_ship_status VARCHAR2(20);
    v_anchorage_status VARCHAR2(20);
BEGIN
    -- Check ship status in the ship table
    SELECT status INTO v_ship_status
    FROM ship
    WHERE ship_id = :NEW.ship_id
      AND company_id = :NEW.company_id;

    IF v_ship_status = 'ACTIVE' THEN
        -- Update ship status to 'Service'
          -- Check anchorage status in the anchorage table
    SELECT status INTO v_anchorage_status
    FROM anchorage
    WHERE id = :NEW.anchorage_id;
          IF v_anchorage_status = 'empty' THEN
        -- Update anchorage status to 'Inactive'
        UPDATE anchorage
        SET status = 'filled'
        WHERE id = :NEW.anchorage_id;

        UPDATE ship
        SET status = 'Service'
        WHERE ship_id = :NEW.ship_id
          AND company_id = :NEW.company_id;
```

```
    ELSE
        -- Raise an exception if anchorage status is not 'Empty'
        RAISE_APPLICATION_ERROR(-20002, 'Anchorage must be
empty to proceed.');
    END IF;


    ELSE
        -- Raise an exception if ship status is not 'Active'
        RAISE_APPLICATION_ERROR(-20001, 'Ship status must be
"Active" to proceed.');
    END IF;




END;
```

Output:



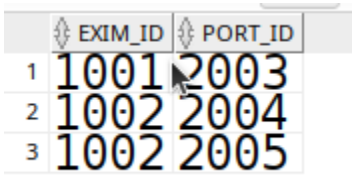
	S_ID	SHIP_ID	COMPANY_ID	CAPTAIN_ID	ANCHORAGE_ID	TO_LOCATION	ANCHORAGE_TIME	DEPT_TIME	STATUS
1	1001	1001	1001	1001	a1	2001	16-NOV-23 01.10.55.352049000 PM	12-DEC-24 12.30.00.000000000 PM	AVAILABLE
2	1002	1002	1002	1002	a2	2003	16-NOV-23 01.11.21.355972000 PM	12-DEC-24 12.30.00.000000000 PM	unavailable
3	1004	1004	1001	1001	c1	2001	16-NOV-23 01.11.21.386552000 PM	22-NOV-23 12.30.00.000000000 PM	AVAILABLE
4	1005	1002	1001	1002	d2	2003	16-NOV-23 01.11.21.392219000 PM	11-DEC-23 12.30.00.000000000 PM	AVAILABLE
5	1006	1003	1001	1002	c2	2004	16-NOV-23 01.11.21.398996000 PM	18-DEC-23 12.30.00.000000000 PM	AVAILABLE

exim_port table:13

```
create table exim_port (
```

```
        exim_id varchar(50),
        port_id  varchar(50),
        foreign key(exim_id) references exim(id),
        foreign key(port_id) references port(id),
        primary key(exim_id,port_id)


);
```

Output:



	EXIM_ID	PORT_ID
1	1001	2003
2	1002	2004
3	1002	2005

load table:14

```
        CREATE TABLE load (
    sailing_id VARCHAR2(50),
    f_id VARCHAR2(50),
    harbor_pilots_id VARCHAR2(50),
    l_time TIMESTAMP,
    PRIMARY KEY (sailing_id, f_id),
    FOREIGN KEY (sailing_id) REFERENCES sailing(s_id),

    FOREIGN KEY (harbor_pilots_id) REFERENCES harbor_pilots(id)
);


CREATE OR REPLACE TRIGGER check_load_time
BEFORE INSERT ON load
```

```
FOR EACH ROW
DECLARE
   v_max_l_time TIMESTAMP;
   v_neet_weight NUMBER;
   ship_id_t VARCHAR2(50);
   company_id_t VARCHAR2(50);
   v_ship_capacity_weight NUMBER;
   sailing_dept_time TIMESTAMP;
       sailing_status varchar2(50);
BEGIN
   -- Find the max l_time for the specified harbor_pilots_id
   SELECT MAX(l_time) INTO v_max_l_time
   FROM load
   WHERE harbor_pilots_id = :NEW.harbor_pilots_id;

       select status into sailing_status
       from sailing
       where s_id=:NEW.sailing_id;
       if sailing_status == 'AVAILABLE' THEN
   -- Check if the new l_time is greater than the allowed time
   IF v_max_l_time IS NULL OR :NEW.l_time > v_max_l_time +
INTERVAL '2' HOUR THEN
       -- Get the dept_time using s_id from sailing table
       SELECT dept_time INTO sailing_dept_time
       FROM sailing
       WHERE s_id = :NEW.sailing_id;

       -- Check if the new l_time is less than the ship's departure time
       IF :NEW.l_time < sailing_dept_time THEN
          -- Get the neet_weight using f_id from ff table
          SELECT neet_weight INTO v_neet_weight
          FROM ff
          WHERE f_id = :NEW.f_id;

          -- Find the ship_id using sailing_id
```

```
        SELECT ship_id INTO ship_id_t
        FROM sailing
        WHERE s_id = :NEW.sailing_id;

        -- Find the company_id using f_id
        SELECT company_id INTO company_id_t
        FROM ff
        WHERE f_id = :NEW.f_id;

        -- Find the ship's capacity_weight
        SELECT capacity_weight INTO v_ship_capacity_weight
        FROM ship
        WHERE ship_id = ship_id_t
          AND company_id = company_id_t;

        -- Check if capacity_weight - neet_weight > 0
        IF v_ship_capacity_weight - v_neet_weight >= 0 THEN
            -- Update the ship's capacity_weight
            UPDATE ship
            SET capacity_weight = v_ship_capacity_weight -
v_neet_weight
            WHERE ship_id = ship_id_t
              AND company_id = company_id_t;

            -- Delete the row from the ff table based on the inserted f_id

        ELSE
            -- Raise an exception if capacity_weight - neet_weight is
not greater than or equal to 0
            RAISE_APPLICATION_ERROR(-20002, 'Capacity weight
is insufficient for the load.');
        END IF;
    ELSE
        -- Raise an exception if the condition is not met
```

```
            RAISE_APPLICATION_ERROR(-20001, 'New l_time must be
less than the ship departure time.');
        END IF;
    ELSE
        -- Raise an exception if the condition is not met
        RAISE_APPLICATION_ERROR(-20001, 'New l_time must be
greater than (2 hours + max l_time).');
    END IF;
 ELSE
        -- Raise an exception if the condition is not met
        RAISE_APPLICATION_ERROR(-20001, 'this sailing is already
leave..you cannot use it');
    END IF;



END;
```

Output:

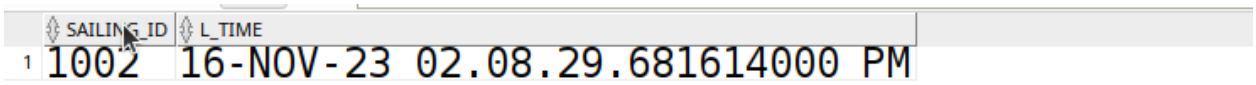
	SAILING_ID	F_ID	HARBOR_PILOTS_ID	L_TIME
1	1006	1006	2001	16-NOV-23 01.37.39.612829000 PM
2	1006	1007	2002	16-NOV-23 01.37.39.628208000 PM
3	1006	1008	2005	16-NOV-23 01.39.00.920721000 PM
4	1006	1009	2006	16-NOV-23 01.39.00.928096000 PM
5	1002	1005	2007	16-NOV-23 01.47.03.963728000 PM

leave table:15

```
create table leave(

        sailing_id varchar(50) primary key,
        l_time  TIMESTAMP,
```

```
        foreign key(sailing_id) references sailing(s_id)
);



create or replace TRIGGER after_leave_insert
AFTER INSERT ON leave
FOR EACH ROW
DECLARE
   v_an VARCHAR2(3);
BEGIN
   UPDATE sailing
   SET status = 'unavailable'
WHERE s_id = :NEW.sailing_id;

   select anchorage_id into v_an
   from sailing
   WHERE s_id = :NEW.sailing_id;

   UPDATE anchorage
   SET status = 'empty'
WHERE id = v_an;

END;
```

output:



M. Finding the Normal Form (1st/2nd/3rd/3+ or BC-NF)

Exim Table:

● There is no transitive dependency.So, this table is in BCNF.

Ship_company:

● There is no transitive dependency.So, this table is in BCNF.

Container:

● There is no transitive dependency.So, this table is in BCNF.

Table: Port:

● There is no transitive dependency.So, this table is in BCNF.

Table: ff

● There is no transitive dependency.So, this table is in BCNF.

Table: Incoming Yard

● There is no transitive dependency.So, this table is in BCNF.

Table: Crane_drivers

● There is no transitive dependency.So, this table is in BCNF.

Table:Harbor_pilots

● There is no transitive dependency.So, this table is in BCNF.

Table:Unload_incoming_yard

● There is no transitive dependency.So, this table is in BCNF.

Table:Ship.
● There is no transitive dependency.So, this table is in BCNF.
Table:Anchorage.
● There is no transitive dependency.So, this table is in BCNF
Table:Sailing
● There is no transitive dependency.So, this table is in BCNF
Table:Exim_port.
● Here every attribute is atomic,and no transitivity. so it is BCNF.

Table:Load.
● There is no transitive dependency.So, this table is in BCNF
Table.Leave.
● There is no transitive dependency.So, this table is in BCNF

N. Future Works and Conclusions

Here ,our port can only perform the export system.In future we will modify its function .We will add import system.Also we will manage how the container should be rearranged.Also we embed servicing center for ship,crane,harbor,truck,container.Also we will manage all the mechanics who are responsible for servicing electronics material.

In conclusion, the Sea Port Management System represents a groundbreaking solution that has significantly enhanced the efficiency and productivity of maritime operations. Through the seamless integration of advanced technologies, such as triggers, stored procedures, and a well-designed database schema, the system has streamlined container handling processes, reducing turnaround times and setting a new standard for operational excellence. The dedication of the development team, coupled with stakeholder support, has enabled the successful navigation of challenges, ensuring the system's robust deployment. Looking ahead, the system is poised for continuous improvement and expansion, with future

works focusing on integrating emerging technologies like IoT for enhanced container tracking and predictive analytics for sailing operations. The Sea Port Management System stands as a testament to our commitment to excellence, fostering innovation and sustainability in the dynamic landscape of maritime logistics.