Design and implementation of database for the shipping company ‘Everblue Ocean Express’

Contents

[Chapter 1 Entity Relationship Diagram and explanation: 2](#_Toc132771341)

[Chapter 2 SQl table definitions and implementation: 6](#_Toc132771342)

[Chapter 3 Multiplicity of relationships between tables: 10](#_Toc132771343)

[Chapter 4 Queries, system requirements and their explanation: 13](#_Toc132771344)

[Chapter 5 Conclusion and Evaluation: 21](#_Toc132771345)

[Credentials for Apex Oracle: 21](#_Toc132771346)

# Chapter 1 Entity Relationship Diagram and explanation:

**ER Diagram:**

A picture containing text, diagram, plan, font

Description automatically generated

**Entities:**

Entities are nothing but tables in the database which are used to store data in rows and columns. There are 7 entities (tables) in this particular design which are:

* Vessel
* Office
* Vesselschedule
* Route
* Container
* Cargo
* Staff

Vessel table records the data into 10 different columns, the columns are

* IMO: This column is unique as it stores an identifying number of each vessel. One of the IMO from the data is 7000001, in which the first 2 digits denote the type of vessel and and last digit denotes the count of the vessel of same type. Here, 70 is type A and 1 is the number of vessel with same type.
* Type- Type stores information on the type of vessel, there are 10 type of vessels with our shipping company each denoted by an alphabet. There are multiple vessels with similar type hence this is not unique.
* Capacity: the capacity is measured in TEU here and one TUE has a capacity of 20ft and weight is around 3000 kilograms. Not unique since a single type of vessels for example ‘A’ have the same capacity.
* Name: Stores the name of each vessel and is unique to all vessels.
* Speed: Denotes the speed of all vessels and its pretty much same for all in database. Hence, not unique.
* Gross Tonnage: This is the overall internal volume of the vessel. This isn’t unique as a single type of vessel will have same gross tonnage.
* Deadweighttonnage: This is total contents of ship including people on it and ship weight. This isn’t unique as a single type of vessel will have same Dead weight tonnage.
* Length: This is the length of the ship, not unique as can be same for same type vessels.
* Breadth: This is the breadth of the vessel, not unique as can be same for same type vessels.
* Date Built: Records the built date of the vessel.

Office table records the data into 5 different columns, the columns are

* Office ID: This column is unique as it stores an identifying number of office.
* Address: This column records the address of office (country in which the office is located). Unique as we assume we have only one office in each country.
* Phone: This column records the phone number of each office. This is unique as each office has different phone number.
* Email: This column records the email of each office. This is unique as each office has different email
* Ports: This records the location or city in which the country’s port is located. This is unique as we assume we have only one port in each office

Vesselschedule table is a relationship table between the vessel and office table. This records the data into 8 columns:

* VesselscheduleID: This column is unique as it stores an identifying number of vessel’s schedule.
* IMO: This column records the ID of each vessel. Foreign key to connect the schedules to vessels in ‘Vessel’ table.
* Vesselfrom: Records the country from which the vessel is starting.
* Fromdate: Record the start date of the vessel from start country.
* Vesselto: Records the country in which the vessel stops.
* Todate: Records the end date of the vessel’s schedule.
* RouteID: Records the routeid which the vessel is associated with. This is a foreign key which connects to route table (which has information of different routes that the shipping company has).
* Officeid: Records the start officeid from which the vessel starts. This is a foreign key which connects the office table.

Route table is a relationship table and it records the route details associated with the shipping company. The columns are as below:

* Routeid: This column is unique as it stores an identifying number of routes.
* Startcountry: The country from which the route starts.
* Startdate: Start date of the route.
* Destination: The country where the route ends.
* Destinationdate: Date where the route ends.
* Stop1: 1st country which comes in between start and destination.
* Stop1date: Date of stop1.
* Stop2: 2nd country which comes in between start and destination.
* Stopdate2: Date of stop2.
* IMO: Vessel id’s associated with routes.

Container table records the data into 18 columns:

* Containerid: This column is unique as it stores an identifying number of containers.
* Name: Records the name of vessel in which the container goes.
* Extlen: External length of the container.
* Extwid: External width of the container.
* Extht: External hieght of the container.
* MGWwt: The total gross weight allowed in the container
* Tarewt: Weight of container without the cargo in it.
* Netwt: Weight of cargo
* Cpickup: Container pickup country.
* Cpickupdate: Container pickupdate.
* Cdrop: Container drop country.
* Cdropdate: Container drop date.
* Intlen: Internal length of the container.
* Intwid: Internal width of the container.
* Intht: Internal hieght of the container.
* Bookingno: Number of cargos assigned to the container.
* Status: Indicated full or available to book. Here we have created constraint that the maximum number of cargos that a container can hold is 3. The customer will not be able to book after this as the field shows full
* IMO: Vessel id’s associated with containers, foreign key to connect the vessels to which the containers are allocated.

Cargo table records the data into 7 columns:

* Cargoid: This column is unique as it stores an identifying number of cargos.
* Containerid: Container id’s associated with cargos
* Goods: Contents of cargo
* Pickup: Cargo pickup date country
* Ddrop: Cargo drop country
* Estimateddropdate: Cargos estimated drop date according to the vessel schedule.
* Isdelivered: Records if the cargo is delivered or not.

Staff table records the data into 5 columns:

* Staffid: This column is unique as it stores an identifying number of the staff.
* Department: This records the department to which the staff belongs to.
* Role: Records the role of the staff.
* Rating: Records the rating given to staff.
* IMO: Vessel id’s associated with staff, foreign key to connect the vessels to which the staff are allocated.

**Relationships:**

The below are the relationships between different entities in this database design:

* Vessel and Office: The relationship is ‘has’. Each vessel has an office and each office has vessels at its ports.
* Vesselschedule and office and vessel: Vesselschedule is a relationship table between office and vessel tables. And the relationship is that ‘at a time’ one vessel is at one office and one office might have multiple vessels in it.
* Vessel and route: The relationship is ‘follows’. Each vessel follows a route.
* Vessel and container: The relationship is ‘has’, vessel has containers.
* Vessel and staff: The relationship is ‘has’, vessel has staff.
* Container and cargo: The relationship is ‘has’. Container has cargo.

**Participation Cardinality:**

* Vessel and office: Vessel is the base table and the relationship between vessel and office is one to many. Each vessel can stop at only one office at a time and each office can have multiple vessels with it at a point of time.
* Vessel and route: This is an assumption that each vessel follows a single route all the time and one route has one vessel following its route. Hence the cardinality is one to one.
* Vessel and container: The cardinality is one to many. Each vessel can have multiple containers but one container goes into only one vessel at a time.
* Vessel and staff: The cardinality is one to many as one vessel can have multiple staff however, one staff member can only belong to one vessel.
* Container and cargo: The cardinality is one to 3 as one container can have 3 cargos as an assumption but one cargo belongs to only one container.

**Constraints:**

* Vessel: IMO is not null and unique and it’s the primary key.
* Office: OfficeID is not null and unique and it’s the primary key.
* Vesselschedule: Vesselscheduleid is not null and unique and it’s the primary key. IMO is not null and foreign key. Routeid is not null and foreign key and Officeid is not null and foreign key.
* Route: RouteID is not null and unique and it’s the primary key. IMO is not null and foreign key.
* Container: ContainerID is not null and unique and it’s the primary key. IMO is not null and foreign key. Bookingno is set to a max 3.
* Cargo: CargoID is not null and unique and it’s the primary key. ContainerID is not null and foreign key.

**Assumptions:**

* Each vessel follows a single route all the time and one route has one vessel following its route.
* The Status column in container table Indicated full or available to book. Here we have created constraint that the maximum number of cargos that a container can hold is 3. The customer will not be able to book after this as the field shows full.
* Only one port in each office
* Only one office in each country.

# Chapter 2 SQl table definitions and implementation:

**Vessel table:**

Code:

Create table Vessel(IMO bigint PRIMARY KEY, Type char(1), Capacity int, Name varchar2(30), Speed int, GrossTonnage int , DeadWeightTonnage int, Length int, Breadth int, DateBuilt date)

A screenshot of a computer

Description automatically generated

**Office Table:**

Code: Create table Office(OfficeID INTEGER PRIMARY KEY GENERATED ALWAYS AS IDENTITY, Address varchar(30), Phone int, email varchar(20), ports varchar(20))

A screenshot of a computer

Description automatically generated

**Route Table:**

Code: Create table Route(RouteID INTEGER GENERATED ALWAYS AS IDENTITY, StartCountry varchar(20), StartDate Date, Destination varchar(20), DestinationDate date, Stop1 varchar(20), Stop1Date date, Stop2 varchar(20), Stop2Date date, imo int NOT NULL, PRIMARY KEY (RouteID), FOREIGN KEY(imo) REFERENCES vessel(imo));

A screenshot of a computer

Description automatically generated

**Vesselschedule table:**

Code: Create table vesselschedule(VesselScheduleID INTEGER GENERATED ALWAYS AS IDENTITY, IMO int NOT NULL, Vesselfrom varchar(20), FromDate Date, vesselto varchar(20),ToDate date,RouteID int,OfficeID int,primary key (VesselScheduleID), foreign key (RouteID) references route(routeid), foreign key (OfficeID) references office(OfficeID))

A screenshot of a computer

Description automatically generated

**Container table:**

Code: Create table container(ContainerID INTEGER GENERATED ALWAYS AS IDENTITY,IMO int NOT NULL,Name varchar(20),ExtLen int,ExtWid int,ExtHt int,MgwWt int,TareWt int,NetWt int,Cpickup varchar(10),CPickupDate date,Cdrop varchar(10),Cdropdate date,Foreign key (IMO) references vessel(IMO))

A screenshot of a computer

Description automatically generated

**Cargo table:**

Code: Create table cargo(CargoID INTEGER GENERATED ALWAYS AS IDENTITY,ContainerID integer,Goods varchar(20),pickup varchar(10),"drop" varchar(10),estimateddropdate date,IsDelivered char(3),foreign key (ContainerID) references container (ContainerID))

A screenshot of a computer

Description automatically generated

**Staff Table:**

Code: Create table staff(StaffID INTEGER GENERATED ALWAYS AS IDENTITY,Department varchar(50),Role varchar(50),Rating varchar(50),IMO number,foreign key (IMO) references vessel (IMO))

A screenshot of a computer

Description automatically generated

# Chapter 3 Multiplicity of relationships between tables:

Vessel and office has one to many relationship each vessel can stop at only one office at a time but one office can have multiple vessels stopping at this is proven in the below figure.

A screenshot of a computer

Description automatically generated

Each vessel follows a single route all the time and one route has one vessel following its route. Hence the relationship is one to one.

A screenshot of a computer

Description automatically generated

The relationship is one to many between vessel and container table. Each vessel can have multiple containers but one container goes into only one vessel at a time.

A screenshot of a computer

Description automatically generated

Vessel and staff has one to many relationship. One vessel can have multiple staff however, one staff member can only belong to one vessel.

A screenshot of a computer

Description automatically generated

Container and cargo: has one to 3 relationship. As one container can have 3 cargos as an assumption but one cargo belongs to only one container.

A screenshot of a computer

Description automatically generated

# Chapter 4 Queries, system requirements and their explanation:

**The below query shows that the details of service routing network is maintained according to the routing of vessels:**

*Code:*

*select vessel.imo, name,startcountry,startdate,destination, destinationdate, stop1,stop1date, stop2, stop2date, routeid*

*from vessel, route*

*where vessel.imo=route.imo order by routeid*

This query fetches the imo number from vessel table, startcountry, start date, destination, destination date, stop1, stop1 date, stop2, stop2date, routeid from route table which shows that the a routing network is maintained with countries to visit and accordingly vessels are assigned to them. Output is as below:

A screenshot of a computer

Description automatically generated

**The below query enables customers to look for sailing schedules:**

*Code:*

*select vesselschedule.imo,vesselscheduleid,vesselschedule.routeid,startdate,destination, destinationdate, stop1,stop1date, stop2, stop2date*

*from vesselschedule, route*

*where vesselschedule.routeid=route.routeid*

*order by vesselscheduleid*

This query fetches imo, vesselscheduleid, routeid from vesselschedule table and startdate,destination, destination date, stop1 and its date and stop 2 and its date from route table which shows that sailing schedule of the vessel including its stops and date of its stops.

A screenshot of a computer

Description automatically generated

**The below query allows the customer to track the cargo:**

*Code:*

*select case when c.ddrop = r.destination then r.destinationdate*

*when c.ddrop = r.stop1 then r.stop1date*

*when c.ddrop = r.stop2 then r.stop2date END ddate*

*from cargo c inner join container con on c.containerid = con.containerid*

*inner join vessel v on v.imo = con.imo*

*inner join route r on v.imo = r.imo*

*where c.cargoid = 10;*

A screenshot of a computer

Description automatically generated

Every customer has their cargo id which uniquely identifies their cargo to be delivered. The above query fetches the delivery date by using case statement which is like a condition that checks the drop location of cargo matches either destination or stop1 or stop 2 of vessel in the route table. So the tables in use: cargo and route are linked using inner join with table container. And in Where condition the customer’s cargoid is given.

**The below query records if the goods are conveyed or not:**

*Code:*

*UPDATE cargo*

*SET cargo.isdelivered = (SELECT CASE*

*WHEN c.estimateddropdate= r.destinationdate OR c.estimateddropdate= r.stop1date OR c.estimateddropdate= r.stop2date THEN 'Yes'*

*ELSE 'No'*

*END AS delivered*

*FROM cargo c inner join container con on c.containerid = con.containerid*

*inner join vessel v on v.imo = con.imo*

*inner join route r on v.imo = r.imo where cargoid = 1) where cargoid = 1*

*A screenshot of a computer

Description automatically generated*

By using update statement, we set the isdelivered column to yes by using case statement where the condition is that the estimated drop date of the cargo is equal to either destination date or stop1 or stop2 date else no. The tables in use route, container, cargo and vessel are connected using multiple inner joins between the tables. If the case is yes then the particular row for the isdelivered column is updated as yes.

**The below query records enables customers to search for containers and book containers:**

The code here is divided into 3 parts:

*Part 1:*

*UPDATE container*

*SET bookingno= (*

*SELECT COUNT(\*)*

*FROM cargo*

*WHERE cargo.containerID = container.containerID)*

*A screenshot of a computer

Description automatically generated*

The above query updates the container table’s bookinngno column with the count of type of container repeats. This basically sees how many times a container is already booked in cargo table and updates container table.

*Part 2:*

*UPDATE container*

*SET status = 'full'*

*WHERE bookingno = 3;*

*A screenshot of a computer

Description automatically generated*

This part of code updates the container table’s status column to full if bookingno from above is 3. This basically allows the viewer to understand if the container is full or not.

*Part 3:*

*CREATE OR REPLACE TRIGGER limit\_record\_insertion*

*BEFORE INSERT ON cargo*

*FOR EACH ROW*

*DECLARE*

*current\_count NUMBER;*

*BEGIN*

*SELECT COUNT(\*) INTO current\_count*

*FROM cargo*

*WHERE containerid = :new.containerid;*

*IF current\_count >= 4 THEN*

*RAISE\_APPLICATION\_ERROR(-20001, 'Maximum number of records reached for this column');*

*END IF;*

*END;*

*A screenshot of a computer

Description automatically generated*

The above query creates a trigger which limits the insertion of cargo ID in turn limits the customer to book the container more than thrice. This is achieved by counting the container id, and if the count of container id greater or equal to 4 it displays an error message that *'Maximum number of records reached for this column'.* Hence restricting any more repetition containerid’s if the count exceeds 3.

**The below query proves that vessel schedules are utilized and the efficiently organized the cargos for transportation of goods:**

*Code:*

*SELECT vesselschedule.vesselscheduleid, vesselschedule.imo, vesselschedule.fromdate, vesselschedule.todate, vesselschedule.vesselfrom, vesselschedule.vesselto,*

*route.startcountry, route.startdate, route.stop1, route.stop1date, route.stop2, route.stop2date, route.destination, route.destinationdate,*

*cargo.cargoid, cargo.pickup, cargo.ddrop, cargo.estimateddropdate*

*FROM container*

*JOIN cargo ON container.containerid = cargo.containerid*

*JOIN vesselschedule ON container.imo = vesselschedule.imo*

*JOIN route ON container.imo = route.imo;*

The query selects the vessel schedule details like schedule id of vessel, its imo, from country and date and to country and date of vessel. The routes start country its stop countries and dates and its destination countries and dates. The cargo’s id, their pickup country and drop country and estimated dates. The output clearly shows that vessel schedule 61 has vessel IMO 7000001 is travelling from 5/1/2023 to 5/6/2023 from China to USA. The associated route’s start country is china on 5/01/2023 with stop1 as South Korea on 05/02/2023 and stop2 Japan on 05/05/2023 and destination USA on 05/06/2023. To this cargo ID I is allocated as it needs to be delivered from China to South Korea and the stop 1 of 7000001 vessel’s route is South Korea with estimated drop day as 05/02/2023.

Note: Dates are on mm/dd/yyyy format and all queries are saved in savedsql option in workspace with File Name : Queries.

# Chapter 5 Conclusion and Evaluation:

By this report I would like to conclude that all the system requirements of maintaining detailed routing network and allocation od schedules to vessels and cargos are done efficiently. Customers are allowed to search for sailing schedules, search for containers and book them. Also, track the cargo accordingly. The model is converted into database, it populated and tested using queries.

The assignment was quite hands on, it helped in applying what I understood during lectures and promoted my critical thinking in learning new concepts in sql as per the design requirements of database. This actually made me quite individual in learning sql and databases. The part that went well was the learning and application of sql concepts. I’ve learned ten times more through the live assignment than what I would regularly learn in the absence of timeline and assignments. The part that didn’t go well was time management. I would manage my time differently segregate my learning and practise hours to catch up the speed of dealing with deadlines but overall it was a very intuitive learning experience.