Starfish Enterprise

Starfish Cruise Line

Database Design Specifications

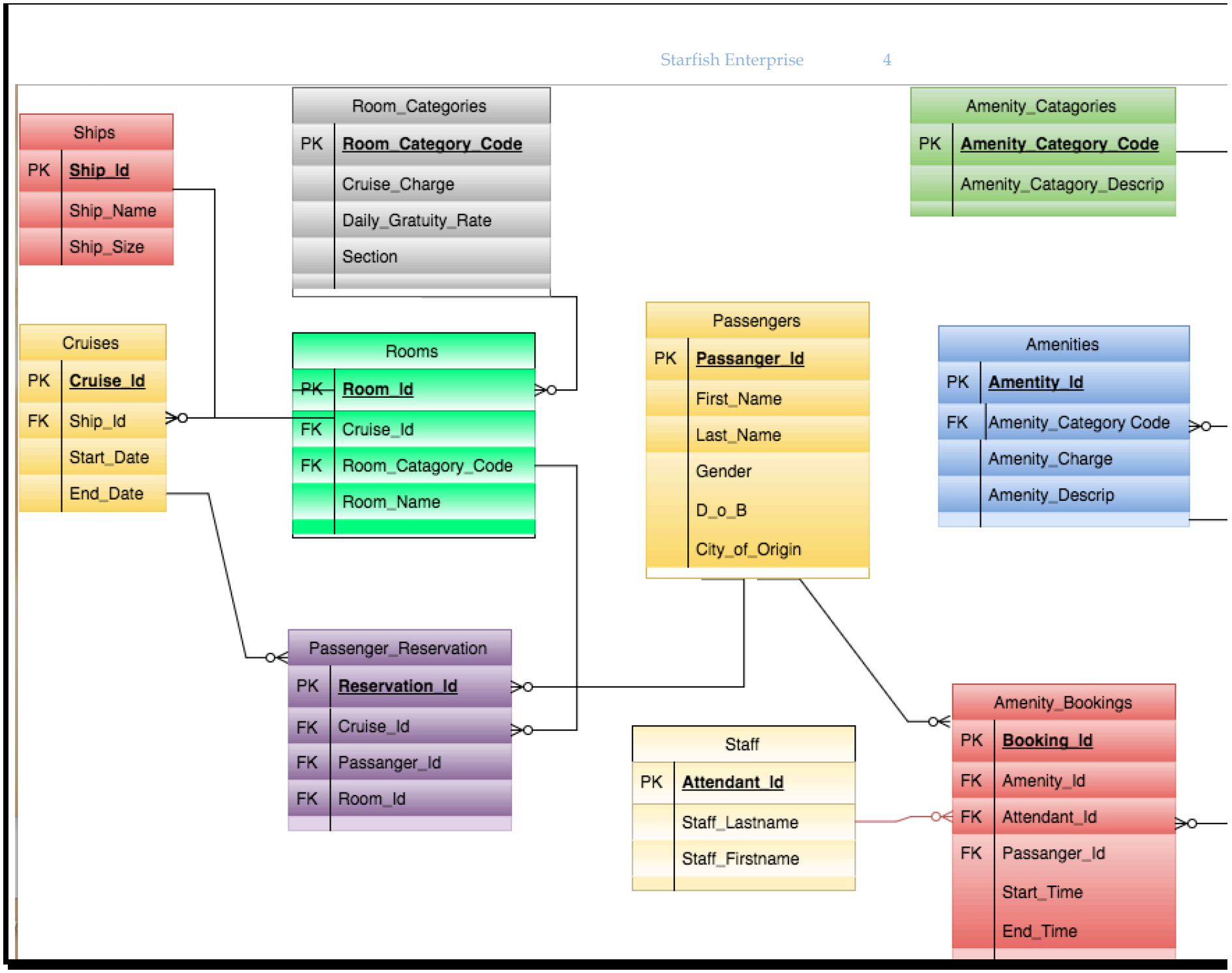
April 19, 2016

Artur Barbosa

Table of Contents

Executive Summary

This document outlines the structure, objects, attributes and entities involved in the design and implementation of a database system for the passengers on a cruise ship. Th purpose of this database is to provide an organized catalog of which ships are used for each specific cruise, along with providing details such as the start date and end date of each cruise. This database will also keep track of which rooms the passengers are in, their amenities, (i.e. room service, phone calls) reservations and staff. This database will help catalog information that will make it easy for administrators to search for necessary data. With the use of this database, passenger activity will be easily accessible, along with many other statistics.


Ships
PK Ship_Id
Ship_Name
Ship_Size
Room_Categories
PK Room_Category_Code
Cruise_Charge
Daily_Gratuity_Rate
Section
Amenity_Catagories
PK Amenity_Category_Code
Amenity_Catagory_Descrip
Cruises
PK Cruise_Id
FK Ship_Id
Start_Date
End_Date
Rooms
PK Room_Id
FK Cruise_Id
FK Room_Catagory_Code
Room_Name
Passengers
PK Passanger_Id
First_Name
Last_Name
Gender
D_o_B
City_of_Origin
Amenities
PK Amentity_Id
FK Amenity_Category Code
Amenity_Charge
Amenity_Descrip
Passenger_Reservation
PK Reservation_Id
FK Cruise_Id
FK Passanger_Id
FK Room_Id
Staff
PK Attendant_Id
Staff_Lastname
Staff_Firstname
Amenity_Bookings
PK Booking_Id
FK Amenity_Id
FK Attendant_Id
FK Passanger_Id
Start_Time
End_Time

Tables

Ships

Purpose

This table is used to store the different ships the cruise line has to offer. It also provides information about the ship, such as the name of the ship and size.

```
CREATE TABLE Ships(
      Ship_ID int NOT NULL,
      Ship_Name varchar(10) NOT NULL,
      Ship_Size varchar(5) NOT NULL,
  PRIMARY KEY(Ship_ID)
);
```

SHIP_ID	SHIP_NAME	SHIP_SIZE
1	Shuana	Sml
2	Albatross	Lrg
3	Cerberus	Lrg
4	Dubbo	Med
5	Arunta	Med

Functional Dependencies

Ship_Id -> Ship_Name, Ship_Size

Cruises

Purpose

This table is used to store each cruise expedition, including data such as what ship was used and when the expedition starts and ends.

```
CREATE TABLE Cruises(
        Cruise_Id int NOT NULL,
        Ship_Id int NOT NULL REFERENCES ships(Ship_ID),
        Start_Date date NOT NULL,
        End_Date date NOT NULL,
PRIMARY KEY(Cruise_Id)
);
```

	CRUISE_ID	SHIP_ID	START_DATE	END_DATE
1	58	1	01-JAN-16	12-JAN-16
2	286	3	02-FEB-16	08-FEB-16
3	102	4	20-APR-16	25-APR-16
4	88	2	15-MAY-16	22-MAY-16

Functional Dependencies

Cruise_Id -> Start_Date, End_Date

Room_Categories

Purpose

The purpose of this table is to store each room on the cruise ship. This table organizes each room by where they are on the ship, how much they should tip for cleaning services, and how much their room is for their entire stay.

```
CREATE TABLE Room_Categories(
    Room_Category_Code varchar(5) NOT NULL,
    Cruise_Charge decimal(6,2) NOT NULL,
    Daily_Gratuity_Rate decimal(5,2) NOT NULL,
    Section varchar(6) NOT NULL,
PRIMARY KEY(Room_Category_Code)
);
```

	ROOM_CATEGORY_CODE	CRUISE_CHARGE	DAILY_GRATUITY_RATE	SECTION
1	11	1162.25	20.5	Cabin
2	22	1950.75	25	Cabin
3	33	2500	32	Lower
4	44	3500.25	35	Upper

Functional Dependencies

Room_Category_Code -> Cruise_charge, Daily_Gratuity_Rate, Section

Rooms

Purpose

The purpose of this table is storing the rooms of the cruise ship. The ID will serve as the room number and it also stores the type of room and which cruise it's on.

```
CREATE TABLE Rooms(
    Room_Id int NOT NULL,
    Cruise_ID int NOT NULL REFERENCES cruises(Cruise_ID),
    Room_Category_Code varchar(5) NOT NULL REFERENCES
room_categories(Room_category_code),
    Room_Name varchar(10) NOT NULL,
PRIMARY KEY (Room_Id)
);
```

Functional Dependencies

Room_Id -> Room_Name

	ROOM_ID	CRUISE_ID	ROOM_CATEGORY_CODE	ROOM_NAME
1	21	58	11	Atlantic
2	101	286	22	Aquarious
3	208	102	22	Aquarious
4	212	58	44	Poseidon

Passenger_Reservation

Purpose

The purpose of this table is to link together information regarding a passenger's reservation. It brings together the passenger id along with the cruise id and room id, which allows the administrator to see details about a passenger's board. All of this information is stored under reservation id.

```
CREATE TABLE Passenger_Reservation(
        Reservation_Id int NOT NULL,
        Cruise_Id int NOT NULL REFERENCES cruises(Cruise_ID),
        Passenger_Id int NOT NULL REFERENCES passengers(Passenger_Id),
        Room_id int NOT NULL REFERENCES rooms(room_id),
PRIMARY KEY (Reservation_Id)
);
```

Functional Dependencies

N/A

	RESERVATION_ID	CRUISE_ID	PASSENGER_ID	ROOM_ID
1	5	58	1	21
2	6	286	3	101
3	7	102	5	208
4	8	58	4	212

Passengers

Purpose

The purpose of this table is to store information regarding the passengers.

```
CREATE TABLE Passengers(
  Passenger_Id int NOT NULL,
      First_Name varchar(10) NOT NULL,
      Last_Name varchar(10) NOT NULL,
      Gender varchar(1) NOT NULL,
      D_O_B date NOT NULL,
      City_of_origin varchar(20) NOT NULL,
PRIMARY KEY(Passenger_Id)
);
```

	PASSENGER_ID	FIRST_NAME	LAST_NAME	GENDER	D_O_B	CITY_OF_ORIGI
1	1	Artur	Barbosa	M	24-JAN-96	Southampton
2	2	Pablo	Rivas	M	20-APR-85	Orlando
3	3	Taylor	Gise	F	19-SEP-95	Tuckahoe
4	4	John	Edwards	M	01-FEB-94	Huntington
5	5	Gloria	Garcia	F	01-MAR-90	Dallas

Functional Dependencies

Passanger_Id -> First_Name, Last_Name, Gender, D_o_B, City_of_Origin

Staff

Purpose

The purpose of this table is to store employees on the ship.

```
CREATE TABLE Staff(
    Attendant_Id int NOT NULL,
    Staff_Lastname varchar(10) NOT NULL,
    Staff_Firstname varchar(10) NOT NULL,
PRIMARY KEY(Attendant_Id)
);
```

	ATTENDANT_ID	STAFF_LASTNAME	STAFF_FIRSTNAME
1	2001	Johnson	Eileen
2	2002	White	Walter
3	2003	Pinkman	Jesse
4	2004	Bryant	Kevin
5	2005	Garcia	Rick
6	2006	Williams	Gerald

Functional Dependencies

Attendant_Id -> Staff_Lastname, Staff_Firstname

Amenity_Catalog

Purpose

The purpose of this table is to store amenities that are available on the cruise ship.

```
CREATE TABLE Amenity_Catalog(
    Amenity_Catagory_Code varchar(4) NOT NULL,
    Amenity_catagory_Descrip varchar(15) NOT NULL,
PRIMARY KEY(Amenity_Category_Code)
);
```

	AMENITY_CAT_CODE	AMENITY_CATAGORY_DESCRIP
1	A01	FoodService
2	A02	Drinks
3	A03	Massage
4	A04	Adult Entertainment
5	A05	Theater Show
6	A06	Spa

Functional Dependencies

Amenity_Catagory_Code -> Amenity_catagory_descrip

Amenities

Purpose

The purpose of this table is store the price and description of amenities ordered by passengers on the ship.

```
CREATE TABLE Amenities(
        Amenity_Id varchar(5) NOT NULL,
        Amenity_Cat_Code varchar(4) NOT NULL REFERENCES
Amenity_Catalog(Amenity_cat_code),
        Amenity_charge decimal(5,2) NOT NULL,
        Amenity_Descrip varchar(16) NOT NULL,
PRIMARY KEY(Amenity_Id)
);
```

	AMENITY_ID	AMENITY_CAT_CODE	AMENITY_CHARGE	AMENITY_DESCRI
1	C01	A02	30.27	Drinks to Room
2	C02	A03	40.25	Room Massage
3	C03	A01	27.85	Food Delivery

Functional Depedencies

Amenity_Id -> Amenity_charge , amenity_descrip

Amenity_Bookings

Purpose

The purpose of this table is to store bookings for amenities (i.e scheduled massage), and which staff member did it.

```
CREATE TABLE Amenity_Bookings(
    Amenity_ID varchar(5) NOT NULL REFERENCES Amenities(Amenity_Id),
    Attendant_Id int NOT NULL REFERENCES Staff(Attendant_ID),
    Passanger_Id int NOT NULL REFERENCES Passengers(Passenger_Id),
    Start_Time time,
    End_Time time,
PRIMARY KEY(Amenity_ID));
```

	BOOKING...	AMENITY...	ATTENDA...	PASSENG...	START_TI...	END_TIME
1	10A	C01	2001	1	7:00	(null)
2	11A	C03	2004	3	9:15	(null)
3	12A	C02	2006	5	14:00	15:00

Functional Dependencies

Amenity_Id -> Start_Time, End_Time

Views

PassengersBoard

Purpose

This view puts together info about the passenger and which room they are staying in th cruise ship if for whatever reasons a staff member needed to know their information.

```
CREATE VIEW PassengersBoard AS
Select P.first_name,
        P.last_name,
        P.gender,
        P.D_o_B,
        R.Room_Id,
        P.Passenger_Id
FROM  passengers P,
          Passenger_Reservation R
WHERE P.passenger_id = R.passenger_id
ORDER BY p.last_name DESC;
```

	FIRST_NAME	LAST_NAME	GENDER	D_O_B	ROOM_ID	PASSENGER_ID
1	Taylor	Gise	F	19-SEP-95	101	3
2	Gloria	Garcia	F	01-MAR-90	208	5
3	John	Edwards	M	01-FEB-94	212	4
4	Artur	Barbosa	M	24-JAN-96	21	1

Reports

Reports

AmenityPopularity

Purpose

The purpose of this view is for the administration of the cruise lines to evaluate the most popular amenities and which ship they were on.

```
CREATE VIEW AmenityPopularity AS
SELECT ships.ship_id,
            ships.ship_name,
            ships.ship_size,
            ab.booking_id,
           ab.amenity_id,
           a.amenity_cat_code
FROM   ships,
            Amenity_Bookings ab,
            Amenities a,
            Passengers P,
            Passenger_Reservation PR,
            Cruises C
WHERE  ab.amenity_id = a.amenity_id
AND      ab.passenger_id = P.Passenger_id
AND     P.Passenger_id = PR.Passenger_id
AND     PR.Cruise_Id = C.Cruise_ID
AND C.Ship_Id = Ships.Ship_Id ;
```

	SHIP_ID	SHIP_N...	SHIP_SIZE	BOOKING_ID	AMENITY_ID	AMENITY_CAT_CODE
1	1	Shuana	Sml	10A	C01	A02
2	3	Cerberus	Lrg	11A	C03	A01
3	4	Dubbo	Med	12A	C02	A03

Staff Activity

Purpose

This query will show the amenities that each staff member served. This will help with monitoring each staff member and how much they are doing on the cruise ship.

```
SELECT ab.attendant_id,
       s.staff_lastname,
       s.staff_firstname,
       ab.Amenity_Id
FROM   Staff s, Amenity_Bookings ab
WHERE  ab.attendant_id = s.attendant_id;
```

	ATTENDANT_ID	STAFF_LASTNAME	STAFF_FIRSTNAME	AMENITY_ID
1	2001	Johnson	Eileen	C01
2	2004	Bryant	Kevin	C03
3	2006	Williams	Gerald	C02

Stored Procedures

Gratuity Total

Purpose This calculates the total gratuity charge for a reservation using the reservation id.

```
CREATE OR REPLACE FUNCTION gratuityTotal(resid int) RETURNS DECIMAL(12,2)
AS $totalcharge$
DECLARE
        charge DECIMAL(12,2);
        stay_total INTEGER;
BEGIN
        stay_total := (SELECT (cruises.end_date - cruises.start_date) AS reservedays
                FROM cruises, passenger_reservation
                WHERE cruises.cruise_id = passenger_reservation.cruise_id
                AND passenger_reservation.reservation_id = resid);

        SELECT INTO charge (room_categories.daily_gratuity_rate * stay_total) AS charge
        FROM cruises, rooms, room_categories, passengers, passenger_reservation
        WHERE passenger_reservation.reservation_id = resid
        AND passenger_reservation.passenger_id = passengers.passenger_id
        AND passenger_reservation.room_id = rooms.roomid
        AND passenger_reservation.cruise_id = cruises.cruise_id
        AND rooms.room_category_code = room_categories.room_category_code;

        RETURN charge;
END;
$totalcharge$
LANGUAGE plpgsql;
```

Room_Total:

Purpose

This calculates the total room charge for a reservation using the reservation id.

```
CREATE OR REPLACE FUNCTION roomTotal(resid int) RETURNS DECIMAL(12,2)
AS $totalcharge$
DECLARE
        charge DECIMAL(12,2);
        stay_total INTEGER;
BEGIN
        stay_total := (SELECT (cruises.end_date - cruises.start_date) AS reservedays
                FROM cruises, passenger_reservation
                WHERE cruises.cruise_id = passenger_reservation.cruise_id
                AND passenger_reservation.reservation_id = resid);
        SELECT INTO charge (room_categories.cruise_charge * stay_total) AS charge
        FROM cruises, rooms, room_categories, passengers, passenger_reservation
        WHERE passenger_reservation.reservation_id = resid
        AND passenger_reservation.passenger_id = passengers.passenger_id
        AND passenger_reservation.room_id = rooms.roomid
        AND passenger_reservation.cruise_id = cruises.cruise_id
        AND rooms.room_category_code = room_categories.room_category_code;

        RETURN charge;
END;
$totalcharge$
LANGUAGE plpgsql;
```

Triggers

RemoveAmenityPassenger

Purpose

This removes amenity bookings on a passenger reservation removal.

```
CREATE OR REPLACE FUNCTION remove_amenity_bookings()
RETURNS trigger AS $$
BEGIN
        DELETE FROM amenity_bookings
        WHERE amenity_bookings.passenger_id = OLD.passenger_id;
        RETURN NULL;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER remove_passenger_amenitybookings
BEFORE DELETE ON passenger_reservations
FOR EACH ROW EXECUTE remove_amenity_bookings();
```

RemoveAmenity

Purpose

This removes amenity bookings on amenity removal.

```
CREATE OR REPLACE FUNCTION remove_amenity()
RETURNS trigger AS $$
BEGIN
        DELETE FROM amenity_bookings
        WHERE amenity_bookings.amenity_id = OLD.amenity_id;
        RETURN NULL;
END;
$$ LANGUAGE plpgsql;


CREATE TRIGGER remove_amenity_amenitybookings
BEFORE DELETE ON amenities
FOR EACH ROW EXECUTE remove_amenity();
```

Security

There are five primary users of the database: passengers, staff, cruise managers and database administrators. For each role, the user will not have any privileges on the tables until they are assigned.

Passengers

Passengers interact with the database indirectly by booking a trip online or over the phone.

GRANT INSERT ON Passengers TO passenger;

Staff

The staff is responsible for booking amenities and scheduling them at appropriate times.

GRANT SELECT, INSERT, UPDATE ON Amenity_Bookings TO Staff;
GRANT SELECT ON Amenity_Catagories TO Staff;
GRANT SELECT ON Amenities TO Saff;

Cruise Managers

Cruise managers are responsible for updating amenities that are available for booking, as well as assigning which staff member is assigned. Cruise Managers must also note which ships are used for each cruise. They are also responsible for updating passenger information and reservations, if needed, along with room reservations.

GRANT SELECT, INSERT, UPDATE ON Amenities TO CruiseManagers;
GRANT SELECT, INSERT, UPDATE ON Staff TO CruiseManagers;
GRANT SELECT, INSERT, UPDATE, DELETE, ON Amenity_Bookings TO CruiseManagers;
GRANT SELECT, INSERT, UPDATE, DELETEON Passenger_Reservation TO CruiseManager;
GRANT SELECT, INSERT, UPDATE ON Passengers TO CruiseManager;
GRANT SELECT, INSERT, UPDATE ON Ships TO CruiseManager;
GRANT SELECT, INSERT, UPDATE ON Ships to Room_Categories;

Database Administrator Role

Has control of everything.

GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public to CruiseAdministrator;

Implementation Notes

The following are suggestions and/or requirements for implementation:

- The datatype for Start_Time and End_Time under Amenity_Bookings is varchar()
- The queries created were not too complex due to the fact that I simply did not have as many passengers and reservations as there would be on a real cruise ship.
- The IDs for passengers, staff and cruise ships were just random numbers when ideally, there would be some sort of format for each.
- Amenity category codes should categorize the amenities (i.e. foodservice, spa would contain cheeseburger and massage, respectively)

Known Problems

- Due to the limited scope, there were many factors I had to omit that would probably be considered in a real cruise liner.
- There is no way to assign two staff members to an amenity in case there are two required.

Future Enhancements

- Creating a table for staff and incorporating their wages into the database.
- Organizing floors and rooms by names.
- Expanding to include resource usage for amenities.
- Expanding amenity table to include gratuity for each amenity.