Copilot

I'm sorry, but I encountered an error while trying to generate SQL queries to answer the questions from the document you uploaded. It seems there was a syntax error in the Python code I used to process the data.

However, I can still provide you with the SQL queries for the questions based on the content of the document:

1. Create the following tables with appropriate domain integrity constraints.

```
CREATE TABLE Sailors(
    sid INTEGER,
    sname VARCHAR(255),
    rating INTEGER,
    age REAL,
    PRIMARY KEY(sid)
);

CREATE TABLE Boats(
    bid INTEGER,
    bname VARCHAR(255),
    color VARCHAR(255),
    PRIMARY KEY(bid)
);

CREATE TABLE Reserves(
    sid INTEGER,
    bid INTEGER,
    day DATE,
    PRIMARY KEY(sid, bid, day),
    FOREIGN KEY(sid) REFERENCES Sailors(sid),
    FOREIGN KEY(bid) REFERENCES Boats(bid)
);
```

1. Insert rows as given below:

```
-- Insert data into Sailors, Boats, and Reserves tables here
```

1. Alter tables Boats and Reserves to establish referential integrity among themselves and the Sailors table.

```
-- The foreign key constraints in the CREATE TABLE statements above establish the referential integrity
```

1. Find the names of sailors who have reserved a red boat.

```
SELECT sname FROM Sailors WHERE sid IN (SELECT sid FROM Reserves WHERE bid IN (SELECT bid FROM Boats WHERE color = 'red'));
```

1. Find the ages of sailors whose name begins with B and has at least three characters.

```
SELECT age FROM Sailors WHERE sname LIKE 'B%' AND LENGTH(sname) >= 3;
```

1. Find the names of sailors who have reserved both a red and a green boat.

```
SELECT sname FROM Sailors WHERE sid IN (SELECT sid FROM Reserves WHERE bid IN (SELECT bid FROM Boats WHERE color = 'red')) AND sid IN (SELECT sid FROM Reserves W
```

1. Find sailors whose rating is better than some sailor called Horatio. (Use ANY / ALL)

```
SELECT sname FROM Sailors WHERE rating > ANY (SELECT rating FROM Sailors WHERE sname = 'Horatio');
```

1. Find the average age of sailors with a rating of 10.

```
SELECT AVG(age) FROM Sailors WHERE rating = 10;
```

1. Find the age of the youngest sailor who is eligible to vote (i.e., is at least 18 years old) for each rating level with at least two such sailors.

```
SELECT rating, MIN(age) FROM Sailors WHERE age >= 18 GROUP BY rating HAVING COUNT(*) >= 2;
```

1. Exemplify Left outer join with respect to sailors and Reserves relations.

```
SELECT * FROM Sailors LEFT OUTER JOIN Reserves ON Sailors.sid = Reserves.sid;
```

1. Create a view named Boat-104 having the sailor-id and day of boat 104.

```
CREATE VIEW Boat_104 AS SELECT sid, day FROM Reserves WHERE bid = 104;
```

1. Show the contents of Boat-104.

```
SELECT * FROM Boat_104;
```

Please note that these queries are based on the assumptions about your database schema and the data you have. You might need to adjust them according to your actual database schema and data. If you have any other questions or need further assistance, feel free to ask! 😊