SELECT: Nested Queries

- Sailors(sid:integer, sname:string,rating:integer,age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- Find the sids of all sailors who have reserved boat 103
  - SELECT S.sname
  - FROM Sailors S, Reserves R
  - WHERE S.sid = R.sid AND R.bid = 103

- Find the sids of all sailors who have reserved boat 103
  - SELECT S.sname
  - FROM Sailors S
  - WHERE S.sid IN (SELECT R.sid
    FROM Reserves R
    WHERE R.bid = 103)
**SELECT: Nested Queries**

- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(sid: integer, bid: integer, day: date)

• Find the sids of all sailors who have not reserved boat 103
  - SELECT S.sid
  - FROM Sailors S
  - WHERE S.sid NOT IN (SELECT R.sid
    FROM Reserves R
    WHERE R.bid = 103)

**SELECT: Nested Queries**

- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(sid: integer, bid: integer, day: date)

• Find the names of sailors who have reserved a red boat
  - SELECT S.sname
  - FROM Sailors S
  - WHERE S.sid IN (SELECT R.sid
    FROM Reserves R
    WHERE R.bid IN (SELECT B.bid
      FROM BOAT B
      WHERE B.color = 'red') )
SELECT: Nested Queries

- Sailors(sid:integer, sname:string,rating:integer,age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

• Find the names of sailors who have not reserved a red boat
  - SELECT S.sname
  - FROM Sailors S
  - WHERE S.sid NOT IN (SELECT R.sid
    - FROM Reserves R
    WHERE R.bid IN (SELECT B.bid
      - FROM BOAT B
      WHERE B.color = 'red'
    )
  )

SELECT: Nested Queries

- Sailors(sid:integer, sname:string,rating:integer,age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

• Find the names of sailors who have reserved boat 103
  - SELECT S.sname
  - FROM Sailors S
  - WHERE EXISTS (SELECT *
    FROM Reserves R
    WHERE R.bid =103 AND R.sid = S.sid )
SELECT: Nested Queries

- Sailors(sid:integer, sname:string,rating:integer,age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- Find the names of sailors who have not reserved boat 103
  - SELECT S.sname
  - FROM Sailors S
  - WHERE NOT EXISTS (SELECT *
      FROM Reserves R
      WHERE R.bid =103 AND R.sid = S.sid )

- Find the sailors whose rating is better than Horatio
  - SELECT S.sid
  - FROM Sailors S
  - WHERE S.rating > ANY (SELECT S2.rating
      FROM Sailors S2
      WHERE S2.sname = 'Horatio' )
SELECT: Nested Queries

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- Find the sailors with the highest rating
  - SELECT S.sid
  - FROM Sailors S
  - WHERE S.rating >= ALL (SELECT S2.rating
  FROM Sailors S2)

- Find the names of sailors who have reserved all boats
SELECT: Nested Queries

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname:string, color:string)
- Reserves(sid:integer, bid:integer, day:date)

- Find the names of sailors who have reserved all boats
  - SELECT S.sname
  - FROM Sailors S
  - WHERE NOT EXISTS (SELECT B.bid
  - FROM Boats B
  - EXCEPT
  - (SELECT R.bid
  - FROM Reserves R
  - WHERE R.sid = S.sid) )

SELECT: Aggregate Operators

- COUNT([DISTINCT] A)
- SUM([DISTINCT] A)
- AVG([DISTINCT] A)
- MAX([DISTINCT] A)
- MIN([DISTINCT] A)

SELECT: Aggregate Operators

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname:string, color:string)
- Reserves(sid:integer, bid:integer, day:date)

- Find the average age of sailors with a rating of 10
SELECT:
Aggregate Operators

• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the average age of sailors with a rating of 10
  – SELECT AVG(S.age)
  – FROM Sailors S
  – WHERE S.rating = 10

SELECT:
Aggregate Operators

• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the name and age of the oldest sailor
  – SELECT S.sname, MAX (S.age)
  – FROM Sailors S
SELECT:
Aggregate Operators

• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the name and age of the oldest sailor
  SELECT S.sname, S.age
  FROM Sailors S
  WHERE S.age = (SELECT MAX(S2.age)
                 FROM Sailors S2)

• Count the number of sailors
  SELECT COUNT (*)
  FROM Sailors S
SELECT:
Aggregate Operators
- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(sid: integer, bid: integer, day: date)
- Count the number of different sailor names

SELECT COUNT (DISTINCT S.sname)
FROM Sailors S
SELECT:
Aggregate Operators
• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the names of sailors who are older than the oldest sailor with a rating of 10
SELECT S.sname
FROM Sailors S
WHERE S.age > (SELECT MAX (S2.age)
FROM Sailors S2
WHERE S2.rating = 10)

SELECT: GROUP BY and HAVING Clauses
• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the age of the youngest sailor for each rating level
SELECT S.rating, MIN (S.age)
FROM Sailors S
GROUP BY S.rating

SELECT: GROUP BY and HAVING Clauses
• Sailors(sid:integer, sname:string,rating:integer,age:real)
• Boats(bid:integer, bname: string, color: string)
• Reserves(sid:integer, bid: integer, day: date)

• Find the age of the youngest sailor
SELECT MIN (S.age)
FROM Sailors S
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- Find the age of the youngest sailor who is eligible to vote, i.e. 18 or older, for each level of rating with at least two such sailors

```
SELECT S.rating, MIN(S.age) AS minage
FROM Sailors S
WHERE S.age >= 19
GROUP BY S.rating
HAVING COUNT(*) > 1

Step 1
```
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

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

```
SELECT S.rating, MIN (S.age) AS minage
FROM Sailors S
WHERE S.age >= 19
GROUP BY S.rating
HAVING COUNT(*) > 1
```

Step 2

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

```
SELECT S.rating, MIN (S.age) AS minage
FROM Sailors S
WHERE S.age >= 19
GROUP BY S.rating
HAVING COUNT(*) > 1
```

Step 3

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

```
SELECT S.rating, MIN (S.age) AS minage
FROM Sailors S
WHERE S.age >= 19
GROUP BY S.rating
HAVING COUNT(*) > 1
```

Step 4
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(sid: integer, bid: integer, day: date)

- Find the age of the youngest sailor who is eligible to vote, i.e., older than 18) for each level of rating with at least two such sailors
  
  SELECT S.rating, MIN(S.age) AS minage
  FROM Sailors S
  WHERE S.age >= 19
  GROUP BY S.rating
  HAVING COUNT(*) > 1

  Step 5

- For each boat, find the number of reservations for the red boat

SELECT: GROUP BY and HAVING Clauses

- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(sid: integer, bid: integer, day: date)

- Find the age of the youngest sailor who is eligible to vote, i.e., older than 18) for each level of rating with at least two such sailors
  
  SELECT S.rating, MIN(S.age) AS minage
  FROM Sailors S
  WHERE S.age >= 19
  GROUP BY S.rating
  HAVING COUNT(*) > 1

  Step 6
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid:integer, sname:string, rating:integer, age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- For each boat, find the number of reservations for the red boat
  SELECT B.bid, COUNT(*) AS reservationcount
  FROM Boats B, Reserve R
  WHERE B.bid = R.bid AND B.color = 'Red'
  GROUP BY B.bid

- Find the average age of sailors for each rating level that has at least two sailors
  SELECT S.rating, AVG(S.age)
  FROM Sailors S
  GROUP BY S.rating
  HAVING COUNT(*) > 1
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid:integer, sname:string,rating:integer,age:real)
- Boats(bid:integer, bname: string, color: string)
- Reserves(sid:integer, bid: integer, day: date)

- Find those ratings for which the average age of the sailors is the minimum over all ratings

```
SELECT S.rating
FROM Sailors S
WHERE AVG(S.age) = (SELECT MIN(AVG(S2.age))
                      FROM Sailors S2
                      GROUP BY S2.rating)
```
SELECT: GROUP BY and HAVING Clauses

- Sailors(sid: integer, sname: string, rating: integer, age: real)
- Boats(bid: integer, bname: string, color: string)
- Reserves(resid: integer, sid: integer, bid: integer, day: date)

- Find those ratings for which the average age of the sailors is the minimum over all ratings

```sql
SELECT Temp.rating, Temp.average
FROM (SELECT S.rating, AVG (S.age) AS average
     FROM Sailors S
     GROUP BY S.rating) AS Temp
WHERE Temp.average = (SELECT MIN(Temp.average)
                      FROM Temp)
```