REM drop tables if exsist.

drop table Reserves;

drop table Sailors;

drop table Boats;

REM create tables accroding to the definition chapter 5.1

create table Sailors(

sid int not null constraint sailors\_pk primary key,

sname varchar2(20),

rating int,

age decimal(4,1)

);

create table Boats(

bid int not null constraint boat\_pk primary key,

bname varchar2(20),

color varchar2(20)

);

create table Reserves(

sid int,

bid int,

day date,

primary key (sid,bid,day),

foreign key (sid) references Sailors(sid)

ON DELETE CASCADE,

foreign key (bid) references Boats(bid)

ON DELETE CASCADE

);

REM initilize the tables as Fig 5.1-5.3, NOTE the order, how about insert reserves before sailors?

insert into Sailors (sid,sname,rating,age)

values(22,'Dustin',7,45.0);

insert into Sailors (sid,sname,rating,age)

values(29,'Brutus',1,33.0);

insert into Sailors (sid,sname,rating,age)

values(31,'Lubber',8,55.5);

insert into Sailors (sid,sname,rating,age)

values(32,'Andy',8,25.5);

insert into Sailors (sid,sname,rating,age)

values(58,'Rusty',10,35.0);

insert into Sailors (sid,sname,rating,age)

values(64,'Horataio',7,35.0);

insert into Sailors (sid,sname,rating,age)

values(71,'Zorba',10,16.0);

insert into Sailors (sid,sname,rating,age)

values(74,'Horataio',9,35.0);

insert into Sailors (sid,sname,rating,age)

values(85,'Art',3,25.5);

insert into Sailors (sid,sname,rating,age)

values(95,'Bob',3,63.5);

insert into Boats (bid,bname,color)

values(101,'Interlake','blue');

insert into Boats (bid,bname,color)

values(102,'Interlake','red');

insert into Boats (bid,bname,color)

values(103,'Clipper','green');

insert into Boats (bid,bname,color)

values(104,'Marine','red');

insert into Reserves(sid,bid,day)

values(22,101,'10-OCT-98');

insert into Reserves(sid,bid,day)

values(22,102,'10-OCT-98');

insert into Reserves(sid,bid,day)

values(22,103,'8-OCT-98');

insert into Reserves(sid,bid,day)

values(22,104,'7-OCT-98');

insert into Reserves(sid,bid,day)

values(31,102,'10-Nov-98');

insert into Reserves(sid,bid,day)

values(31,103,'6-NOV-98');

insert into Reserves(sid,bid,day)

values(31,104,'12-NOV-98');

insert into Reserves(sid,bid,day)

values(64,101,'5-SEP-98');

insert into Reserves(sid,bid,day)

values(64,102,'8-SEP-98');

insert into Reserves(sid,bid,day)

values(74,103,'8-SEP-98');

REM Q1 "Find the names of sailors who have reserved boat number 103"

SELECT S.sname

FROM Sailors S, Reserves R

WHERE S.sid = R.Sid AND R.bid = 103;

REM Nested Query

SELECT S.sname

FROM Sailors S

WHERE S.sid IN (SELECT R.sid

FROM Reserves R

WHERE R.bid = 103);

REM Correlated Nested Queries

SELECT S.sname

FROM Sailors S

WHERE EXISTS (SELECT \*

FROM Reserves R

WHERE R.bid = 103

AND R.sid = S.sid);

REM "Find the names of sailors who have never reserved boat number 103"

REM Which of the following is right?

SELECT S.sname

FROM Sailors S, Reserves R

WHERE S.sid = R.Sid AND R.bid != 103;

SELECT S.sname

FROM Sailors S

WHERE S.sid NOT IN (SELECT R.sid

FROM Reserves R

WHERE R.bid = 103);

REM Q2 "Find the names of sailors who have reserved a red boat"

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = 'red';

REM Nested Query

SELECT S.sname

FROM Sailors S

WHERE S.sid IN (SELECT R.sid

FROM Reserves R

WHERE R.bid IN(SELECT B.bid

FROM Boats B

WHERE B.color ='red'));

REM Q3 "Find the colors of boats reserved by Lubber"

SELECT B.color

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND S.sname ='Lubber';

REM Q4 "Find the names of sailors who have reserved at least one boat"

SELECT S.sname

FROM Sailors S, Reserves R

WHERE S.sid = R.sid ;

SELECT DISTINCT S.sname

FROM Sailors S, Reserves R

WHERE S.sid = R.sid;

REM Q5 "Find the names of sailors who have reserved a red or a gree boat"

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid=B.bid AND (B.color='red' OR B.color='green');

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color='red'

UNION

SELECT S2.sname

FROM Sailors S2, Boats B2, Reserves R2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid AND B2.color = 'green';

REM Q6 "Find the names of sailors who have reserved both a red and a green boat"

SELECT S.sname

FROM Sailors S, Reserves R1, Boats B1, Reserves R2, Boats B2

WHERE S.sid = R1.sid AND R1.bid = B1.bid

AND S.sid = R2.sid AND R2.bid = B2.bid

AND B1.color = 'red' AND B2.color = 'green';

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = 'red'

INTERSECT

SELECT S2.sname

FROM Sailors S2, Boats B2, Reserves R2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid AND B2.color= 'green';

REM Nested Query

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color ='red'

AND S.sid IN (SELECT S2.sid

FROM Sailors S2, Boats B2, Reserves R2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid

AND B2.color ='green');

REM Q "Find the names of sailors who have reserved a red but not a gree boat"

REM Oracle does not have EXCEPT but use keyword MINUS.

REM when we run the examples in our textbook, we should change EXCEPT to MINUS.

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = 'red'

MINUS

SELECT S2.sname

FROM Sailors S2, Boats B2, Reserves R2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid AND B2.color= 'green';

SELECT S.sname

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color ='red'

AND S.sid NOT IN (SELECT S2.sid

FROM Sailors S2, Boats B2, Reserves R2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid

AND B2.color ='green');

REM Q7 "Find the names of sailors who have reserved at least two different boats"

SELECT DISTINCT S.sname

FROM Sailors S, Reserves R1, Reserves R2

WHERE S.sid = R1.sid AND R1.sid = R2.sid AND R1.bid != R2.bid;

REM "Find the names of sailors who have reserved at least n boats"

REM THE SAME IDEA IS TO JOIN N RELATIONS --- TOO DEDIOUS

REM We can do this by combining CNT, GROUP BY, and nested query together.

REM The question is how we can do this before we adress GROUP BY.

REM Assume one dbms does not support GROUP BY and HAVING, how will you help

REM them implement this? HINT: the same relation equ-join many times.

REM IS THERE ANY DIFFERENCE BETWEEN THE TWO FOLLOWING EXPRESSION?

REM IS the next one the same as the above one?

REM INSERT INTO Reserves Values(74,103,'08-DEC-98');

SELECT S.sname

from Sailors S, Reserves R

where S.sid = R.sid

GROUP BY S.sname

HAVING COUNT(\*) > 1;

SELECT S1.sname

FROM Sailors S1

WHERE S1.sid IN (

SELECT S.sid

from Sailors S, Reserves R

where S.sid = R.sid

GROUP BY S.sid

HAVING COUNT(\*) > 1);

REM IF YOU RUN THE TWO EXPRESSION OVER THE CURRENT INSTANCE, NO DIFFERENCE BETWEEN THE RESULT

REM HOW ABOUT WE INSERT TWO NEW TUPLES, CHECK THE DIFFERENCE.

REM insert into Sailors (sid,sname,rating,age)

REM values(131,'Lubber',8,55.5);

REM insert into Reserves(sid,bid,day)

REM values(131,101,'8-OCT-98');

REM Q8 "Find the sids of sailors with age over 20 who have not reserved a red boat"

REM Q9 "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 )

MINUS

(SELECT R.bid

FROM Reserves R

WHERE R.sid = S.sid));

REM HINT: for each sailor we check that there is no boat that has not been reserved by this sailor

SELECT S.sname

FROM Sailors S

WHERE NOT EXISTS (SELECT B.bid

FROM Boats B

WHERE NOT EXISTS(SELECT R.bid

FROM Reserves R

WHERE R.bid = B.bid

AND R.sid = S.sid));

REM Q10 "Find the names of sailors who have reserved all boats called Interlake"

SELECT S.sname

FROM Sailors S

WHERE NOT EXISTS (SELECT B.bid

FROM Boats B

WHERE B.bname ='Interlake' AND

NOT EXISTS(SELECT R.bid

FROM Reserves R

WHERE R.bid = B.bid

AND R.sid = S.sid));

REM Q11 "Find all sailors with a rating above 7"

SELECT S.sid, S.sname, S.rating, S.age

FROM Sailors S

WHERE S.rating > 7;

REM Q12 "Find the names and ages of sailors with a rating above 7"

SELECT S.Sname, S.age

FROM Sailors S

WHERE S.rating > 7;

REM Q13 "Find the sailor name boat id and reservation date for each reservation"

REM Q14 "Find sailors who have reserved all red boats"

REM Q15 "Find the names and ages of all sailors"

SELECT DISTINCT S.sname, S.age

FROM Sailors S;

REM Q16 "Find the sids of sailors who have reserved a red boat";

SELECT R.sid

FROM Boats B, Reserves R

WHERE B.bid = R.bid AND B.color = 'red';

REM Q17 "Compute increments for the ratings of persons who have sailed two different boats on the same day"

SELECT S.sname, S.rating +1 AS rating

FROM Sailors S, Reserves R1, Reserves R2

WHERE S.sid = R1.sid AND S.sid = R2.sid

AND R1.day = R2.day AND R1.bid <> R2.bid;

REM Q18 Find the ages of sailors whose name begins and ends with B and has at least three characters

SELECT S.age

FROM Sailors S

WHERE S.sname LIKE 'B\_%B';

REM Q19 Find the sids of all sailors who have reserved red boats but not green boats

SELECT S.sid

FROM Sailors S, Reserves R, Boats B

WHERE S.sid = R.sid AND R.bid = B.bid AND B.color = 'red'

MINUS

SELECT S2.sid

FROM Sailors S2, Reserves R2, Boats B2

WHERE S2.sid = R2.sid AND R2.bid = B2.bid AND B2.color = 'green';

SELECT R.sid

FROM Boats B, Reserves R

WHERE R.bid = B.bid AND B.color = 'red'

MINUS

SELECT R2.sid

FROM Boats B2, Reserves R2

WHERE R2.bid = B2.bid AND B2.color = 'green';

REM Q20 "Find all sids of sailors who have a rating of 10 or have reserved boat 104"

SELECT S.sid

FROM Sailors S

WHERE S.rating = 10

UNION

SELECT R.sid

FROM Reserves R

WHERE R.bid = 104;

REM Q21 "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 Boats B

WHERE B.color='red'));

REM Q22 "Find sailors whose rating is better than some sailor called Horatio"

REM SET comparsion operators

SELECT S.sid

FROM Sailors S

WHERE S.rating > ANY(SELECT S2.rating

FROM Sailors S2

WHERE S2.sname = 'Horatio');

REM Q23 "Find sailors whose rating is better than every sailor called Horatio"

SELECT S.sid

FROM Sailors S

WHERE S.rating > ALL(SELECT S2.rating

FROM Sailors S2

WHERE S2.sname = 'Horatio');

REM Q24 "Find the sailors with the highest rating"

SELECT S.sid

FROM Sailors S

WHERE S.rating >= ALL(SELECT S2.rating FROM Sailors S2);

REM Q25 "Find the average of all sailors"

SELECT AVG(S.age)

FROM Sailors S

REM Q26 "Find the average age of sailors with a rating of 10"

SELECT AVG(S.age)

FROM Sailors S

WHERE S.rating = 10;

REM Q27 "Find the name and age of the oldest sailor"

REM This query is illegal in SQL-- of the SELECT clause uses an aggregate operation, then it must use

REM only aggregate operations unless the query contains a GROUP By clause.

SELECT S.sname, MAX (S.age)

FROM Sailors S;

SELECT S.sname, S.age

FROM Sailors S

WHERE S.age = (SELECT MAX(S2.age)

FROM Sailors S2);

REM the following is legal in SQL92 but may not be supported in many systems

REM Oracle support

SELECT S.sname, S.age

FROM Sailors S

WHERE (SELECT MAX(S2.age)

FROM Sailors S2) = S.age ;

REM Q28 "Count the number of sailors"

SELECT COUNT(\*)

FROM Sailors S;

REM Q29 "Count the number of different sailor names"

SELECT COUNT (DISTINCT S.sname)

FROM Sailors S;

REM Q30 "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 S.sname

FROM Sailors S

WHERE S.age > ALL (SELECT S2.age

FROM Sailors S2

WHERE S2.rating = 10);

REM Q31 "Find the age of the youngest sailor for each rating level"

REM SELECT MIN(S.age)

REM FROM Sailors S

REM WHERE S.rating = i;

SELECT S.rating, MIN(S.age)

FROM Sailors S

GROUP BY S.rating ;

REM Q32 "Find the age of the youngest sailor who is eligible to vote(i.e., is at least 18 years old)

REM for each rating level with at least two such sailors"

SELECT S.rating, MIN(S.age) AS minage

FROM Sailors S

WHERE S.age >=18

GROUP BY S.rating

HAVING COUNT(\*) > 1;

REM Q33 "For each red boat, find the number of reservations for this boat"

SELECT B.bid, COUNT(\*) AS sailorcount

FROM Boats B, Reserves R

WHERE R.bid = B.bid AND B.color = 'red'

GROUP BY B.bid;

REM the following is illegal for only columns that appear in the group by clause can appear in the

REM having clause, unless they appear as arguments to an aggregate operator in the

REM having clause.

SELECT B.bid, COUNT(\*) AS sailorcount

FROM Boats B, Reserves R

WHERE R.bid = B.bid

GROUP BY B.bid

HAVING B.color ='red' ;

REM Q34 "Find the average age of sailors for each rating level that has at least two sailors"

SELECT S.rating, AVG(S.age) AS average

FROM Sailors S

GROUP BY S.rating

HAVING COUNT(\*) > 1;

SELECT S.rating, AVG (S.age) AS average

FROM Sailors S

GROUP BY S.rating

HAVING 1 < (SELECT COUNT(\*)

FROM Sailors S2

WHERE S.rating = S2.rating);

REM Q35 "Find the average age of sailors who are of voting age(i.e., at least 18 years old)

REM for each rating level that has at least two sailors"

SELECT S.rating, AVG(S.age) AS average

FROM Sailors S

WHERE S.age >=18

GROUP BY S.rating

HAVING 1 <(SELECT COUNT(\*)

FROM Sailors S2

WHERE S.rating = S2.rating);

REM Q36 "Find the average age of sailors who are of voting age(i.e., at least 18 years old)

REM for each rating level that has at least two such sailors"

SELECT S.rating, AVG(S.age) AS average

FROM Sailors S

WHERE S.age > 18

GROUP BY S.rating

HAVING 1<(SELECT COUNT(\*)

FROM Sailors S2

WHERE S.rating=S2.rating AND S2.age>=18);

SELECT S.rating, AVG(S.age) AS average

FROM Sailors S

WHERE S.age > 18

GROUP BY S.rating

HAVING COUNT(\*) > 1;

REM Q37 "Find those rating sfor which the average age of sailors in the minimum over all ratings"

REM SELECT S.rating

REM FROM Sailors S

REM WHERE AVG(S.age) = (SELECT MIN(AVG(S2.age))

REM FROM Sailors S2

REM GROUP BY S2.rating)

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);

REM How about the following, is it the same as the above?

SELECT Temp.rating, MIN(Temp.average)

FROM (SELECT S.rating, AVG(S.age) AS average,

FROM Sailors S

GROUP BY S.rating) AS Temp

GROUP BY Temp.rating;

REM One student figures out the following approach as all the above do not work in oracle.

REM-----------------------------------------------------------

CREATE VIEW Temp(Rating,Average) AS

(SELECT S.rating, AVG(S.age)

FROM Sailors S

GROUP BY S.rating) ;

SELECT Temp.Rating,Temp.Average

FROM Temp

WHERE Temp.Average = (SELECT MIN(Temp.Average) FROM Temp) ;

drop view Temp;

REM --------------------------------------------------------------

REM New Query "find the sailor ids with top 5 rating ranks."

SELECT sid, rating

FROM ( SELECT sid, rating, RANK() OVER (ORDER BY rating DESC) rating\_rank FROM sailors )

WHERE rating\_rank <= 5;