Kevin Kannappan

Winter 2019

DSE201: Final Exam

**Problem 1:**

1. **False**. If you are sub-querying, aliases are always required even if not necessary with original joins in the schema.
2. **False**. Non-numeric data or nulls on the numeric data.
3. **True.** Many ways to do this, primary would be to use a union function with match and non-match.
4. **True.** Can use an aggregating function.
5. **True.** You can transform the minimum function OR you could also use the GREATEST function, which is available in Postgresql.
6. **True.** Considering it is a **non-null** foreign key that references S, there will be no tuples that will be removed by creating this condition.
7. **True.** Replace except with a NOT EXISTS function
8. **False.** The results of a sub-condition query referencing multiple tables cannot be unnested without created some sort of table store (view, temp table, etc.)
9. **False.** The syntax of the first statement will not execute properly.
10. **True.** Considering that A is the primary key (i.e. no duplicate A), and there is a non-null, numeric B. Then there must be 1 and only 1 value of B for each A, hence the aggregating Max function will return the same original table R.

**Problem 2:**

-- Part i

select sum(case when (hteam like '%San Diego%' and hscore > vscore)

OR (vteam like '%San Diego%' and vscore > hscore)

then 1 else 0 end) as wins

from matches

-- Part ii

select a.team, sum(points) as points

from

(select hteam as team, sum(case when hscore > vscore then 2

when hscore = vscore then 1 else 0 end)

as points

from matches

group by 1

union all

select vteam as team, sum(case when hscore < vscore then 3

when hscore = vscore then 1 else 0 end)

as points

from matches

group by 1) a

group by 1

-- Part iii

select coach from teams

where name IN(

select distinct team from

(select hteam as team, sum(case when hscore >= vscore then 1 else 0 end) as win\_draw, count(\*) as games

from matches

group by 1

having count(\*) = sum(case when hscore >= vscore then 1 else 0 end)

union all

select vteam as team, sum(case when vscore >= hscore then 1 else 0 end) as win\_draw, count(\*) as games

from matches

group by 1

having count(\*) = sum(case when vscore >= hscore then 1 else 0 end)) a

)

-- Part iv

-- Create leaderboard table

CREATE VIEW leaders(team, points, standing) as

select aa.\*, RANK() OVER (order by points desc) as standing

from (select a.team, sum(points) as points

from

(select hteam as team, sum(case when hscore > vscore then 2

when hscore = vscore then 1 else 0 end)

as points

from matches

group by 1

union all

select vteam as team, sum(case when hscore < vscore then 3

when hscore = vscore then 1 else 0 end)

as points

from matches

group by 1) a

group by 1)aa;

select name

from teams

where name NOT IN (

select distinct t.name

from teams t, matches m

where t.name = m.hteam

and m.vscore > m.hscore

and m.vteam NOT IN (

select team

from leaders

where standing <= 1)

UNION ALL

select t.name

from teams t, matches m

where t.name = m.vteam

and m.hscore > m.vscore

and m.hteam NOT IN (

select team

from leaders

where standing <= 1));

-- Part v

-- Could make the following indexes:

CREATE INDEX matches\_hteam\_index ON matches(hscore);

CREATE INDEX matches\_vteam\_index ON matches(vscore);

-- These indexes may help once many seasons of matches are inputted into the table

-- right now, however, the indexes may not provide much of a lift.

-- Both of these selections were made because we aggregate heavily on both

-- of those fields to yield the metrics/data we care about. The scores are counted

-- directly, and do not appear to be aggregated on, hence I felt that an

-- index on both would be of lesser importance.

-- Part vi

CREATE TRIGGER scoreboard\_team

AFTER INSERT ON teams

for each row

BEGIN

INSERT INTO scoreboard VALUES (:new.name, 0);

END;

CREATE TRIGGER scoreboard\_updater

AFTER INSERT ON matches

for each row

BEGIN

IF :new.hscore < :new.vscore

THEN

UPDATE scoreboard

SET points = points + 3

where name = :new.vteam;

ELSEIF :new.hscore > :new.vscore

THEN

UPDATE scoreboard

SET points = points + 2

where name = :new.hteam;

ELSE

UPDATE scoreboard

SET points = points + 1

where name = :new.vteam;

UPDATE scoreboard

SET points = points + 1

where name = :new.hteam;

END IF;

END;