130 Intro to Database 2020 Exam Sample Solution

Disclaimer: This is not an official answer key, so please correct any mistake if you find one. There are more than one possible solution for some questions, so keep in mind about that!

1. **a.**

$\pi_{\text{abbreviation, established, country}}\sigma_{\text{organization.abbreviation=is\_member.organization}\wedge\text{type='member'}}(\text{organization} \times \text{is\_member})$

b. i) $(\pi_{\text{name}}\text{country}) - (\pi_{\text{name}}\sigma_{\text{country.code=is\_member.country}\wedge\text{type='member'}}(\text{country} \times \text{is\_member}))$

ii) $\text{not\_a\_member\_of\_any(Name)}$:-

$\text{country(Name, Code, \_, \_, \_)}$

$\neg\ \text{is\_member(Code, \_, 'member')}$

iii) A possible solution:

```sql
SELECT name
FROM country
EXCEPT
SELECT name
FROM country JOIN is_member ON country.code = is_member.country
WHERE is_member.type = 'member'
```

c. A possible solution:

```sql
SELECT c.name AS name, c.population AS population,
       (SELECT SUM(b.length)
        FROM borders AS b
        WHERE b.countr1 = c.name
        OR b.country2 = c.name) AS border_length
FROM country AS c
```

Trying this by sshing into Postgres databases didn't give any values for border_length. But my code seems to work (although I do not know if the JOIN...OR... syntax is ANSI SQL):

```
SELECT name,
       population,
       SUM(length) AS border_length
FROM   country JOIN borders
       ON country1=code
       OR country2=code
GROUP BY name, population;
```

d. i) Return the abbreiviations of the organizations whose full members all have areas greater than 40,000. Result: (5 rows) WFTU, C, AL, CSTO, PCA. *(Thanks to Kaiyan Fan for noticing that the organization whose abbreviation does not appear in the is_member table should also be returned)*

ii) Two possible solutions:

```sql
--Version 1
SELECT DISTINCT is_member.organization
FROM is_member JOIN country ON is_member.country = country.code
WHERE country.area > 40000
OR is_member.type = 'member'
```

This does not return the correct answers when tested on the database. e.g. NATO and EU are a part of the table which this query returns

```sql
--Version 2: Thanks to Kaiyan Fan
SELECT organization.abbreviation
FROM organization
WHERE 40000 < ALL (SELECT country.area
                   FROM is_member JOIN country ON is_member.country =
country.code
                   WHERE is_member.organization = organization.abbreviation
                   AND is_member.type = 'member')
```

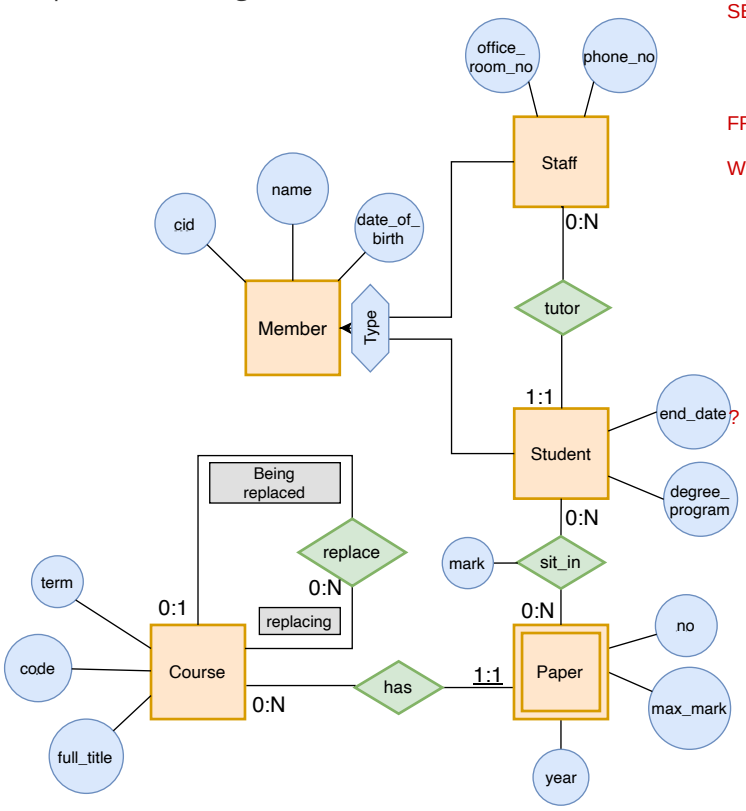
e. A possible (overcomplicated) solution:

```sql
SELECT m.organization AS organization, c.name AS name, c.population AS
population,
       (SELECT c.population / SUM(c2.population)
        FROM country AS c2 JOIN is_member AS m2 ON m2.country = c2.code
                                                AND m2.organization =
m.organization)
       AS pc_total
FROM country AS c JOIN is_member AS m ON m.country = c.code
WHERE m.type = 'member'
```

The solution for (e) does not seem to work at all giving 0 in all rows for pc_total. I think this solution does work however:

```sql
SELECT organization,
       name,
       population,
       (population * 100.0 / (SUM (population) OVER
           (PARTITION BY organization))) AS pc_total
FROM    is_member JOIN country
       ON country.code=is_member.country
WHERE  type='member';
```

2. **a. i)** A possible ER diagram is shown below:



(Thanks to Fankai Yan for noticing that Paper should include the property "year")

ii) A possible relational schema is shown below:

$\text{member}(\underline{\text{cid}}, \text{name}, \text{date\_of\_birth})$
$\text{staff}(\underline{\text{cid}}, \text{office\_room\_no}, \text{phone\_no})$
$\text{student}(\underline{\text{cid}}, \text{degree\_program}, \text{end\_date}, \text{tutor\_cid})$
$\text{staff}(\text{cid}) \overset{\text{fk}}{\Rightarrow} \text{member}(\text{cid})$ $\text{student}(\text{cid}) \overset{\text{fk}}{\Rightarrow} \text{member}(\text{cid})$ $\text{student}(\text{tutor\_cid}) \overset{\text{fk}}{\Rightarrow} \text{staff}(\text{cid})$
$\text{course}(\underline{\text{code}}, \text{full\_title}, \text{term})$ $\text{replace}(\underline{\text{replacing\_code}}, \underline{\text{being\_replaced\_code}})$
$\text{replace}(\text{replacing\_code}) \overset{\text{fk}}{\Rightarrow} \text{course}(\text{code})$ $\text{replace}(\text{being\_replaced\_code}) \overset{\text{fk}}{\Rightarrow} \text{course}(\text{code})$
$\text{paper}(\underline{\text{code}}, \underline{\text{no}}, \text{max\_mark}, \text{year})$ $\text{paper}(\text{code}) \overset{\text{fk}}{\Rightarrow} \text{course}(\text{code})$
$\text{sit\_in}(\underline{\text{student\_cid}}, \underline{\text{paper\_no}}, \text{mark})$
$\text{sit\_in}(\text{student\_cid}) \overset{\text{fk}}{\Rightarrow} \text{student}(\text{cid})$ $\text{sit\_in}(\text{paper\_no}) \overset{\text{fk}}{\Rightarrow} \text{paper}(\text{no})$

b. i) Since $A \to E$, $AE \to F \Rightarrow A \to F$.

We can delete $BD \to B$ as well since it's trivial, leaving us $BD \to G$.

We can delete $C \to G$ as it's trivial as well, leaving us $C \to AF$.

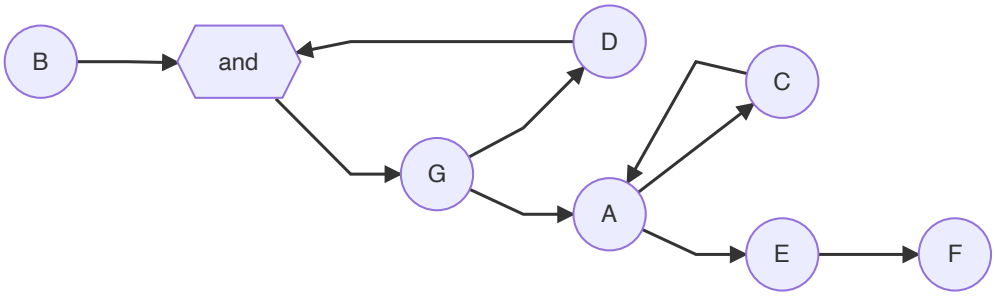
Since $C \to A$, $A \to F$, we have $C \to F \Rightarrow \emptyset$.

Since $A \to E$, $E \to F$, we have $A \to F \Rightarrow \emptyset$.

Hence, the minimal cover S_c is $\{BD \to G, G \to AD, A \to CE, C \to A, E \to F\}$

ii) B must be in all candidate keys since no FD has the form $X \to B$ that helps us to get B.

The candidate keys are BD and BG, according to the folloiwng reachability graph.



iii) The relation is not yet in 3NF since $E \to F$ does not satisfy 3NF requirement when projecting on R_2 . Decomposing R_2 using $R \to F$ will yield $R_4(A, C, E)$ and $R_5(E, F)$. As a result, R_1, R_3, R_4, R_5 are all in 3NF.

c. i) H_a is not serialisable since a lost update occurred: $r_1[C_R] \prec w_3[C_R] \prec w_1[C_R]$. In this case, H_1 will overwrite C_R after update by H_3

H_a is recoverable because neither H_3 nor H_1 reads object written by the other transaction before their commit.

ii) H_b is not serialisable since an inconsistent analysis occurred: $w_3[C_R] \prec r_2[C_R], r_2[C_B] \prec w_3[C_B]$.

H_b is not recoverable because although $r_1[C_R]$ reads from $w_3[C_R]$ and H_1 commits before H_3.

Notice that a dirty read has occurred as well: $w_3[C_R] \prec r_2[C_R] \prec c_3$.