CSC 336 Project

Cristian Statescu

Professor Hesham Auda

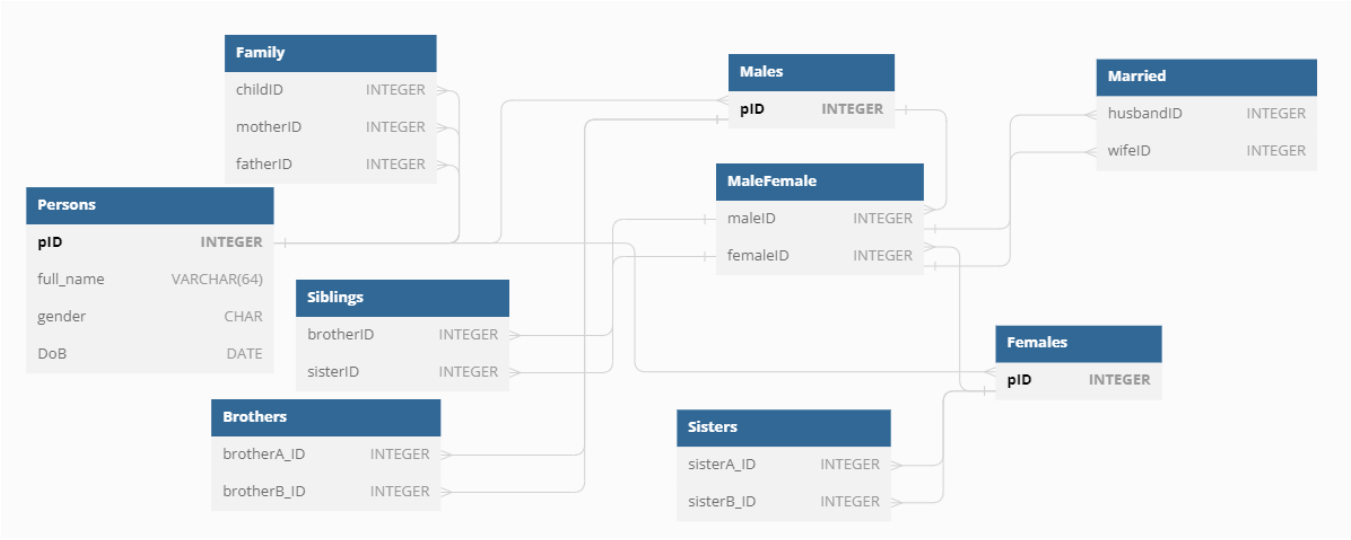
Part A

Question:

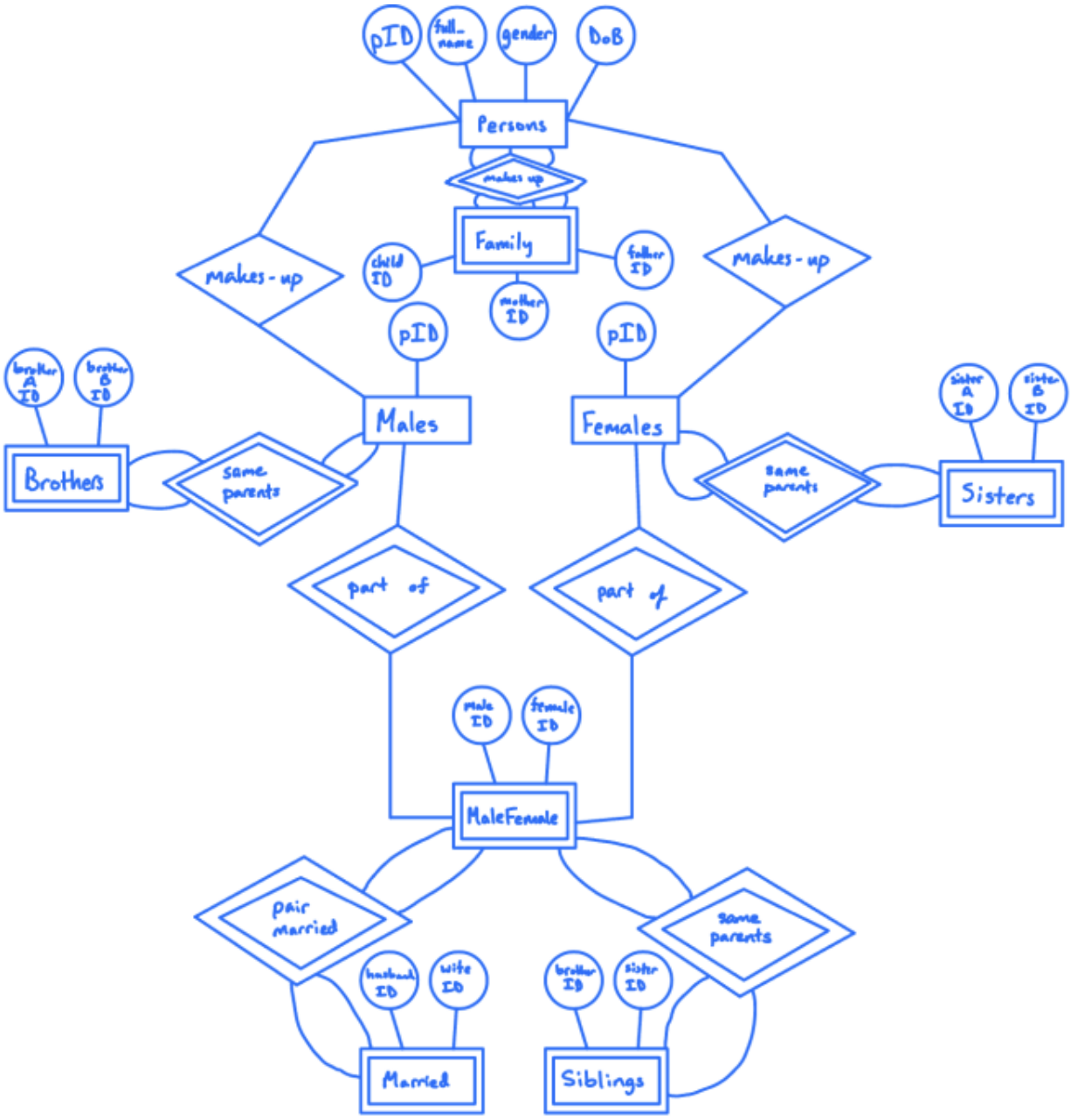
Given the relation PERSONS that has tuples holding the attributes of persons (p), and the relation FAMILY that has tuples of the form (pId, fId, mId), where pId is the Id person in PERSONS, fId is the Id of the father of p, and mId is the Id of the mother of p, with f and m are also persons in PERSONS.

Provide the E-R diagram and corresponding relational database schema used for this project. Indicate for each relation: the key or keys, primary key, foreign keys, essential constraints, and any appropriate checks.

Schema:



ER Diagram:


pID
full_name
gender
DoB
Persons
makes up
Family
child ID
father ID
mother ID
makes - up
makes - up
pID
pID
brother A ID
brother B ID
Males
Females
sister A ID
sister B ID
Brothers
same parents
same parents
Sisters
part of
part of
male ID
female ID
MaleFemale
pair married
same parents
husband ID
wife ID
brother ID
sister ID
Married
Siblings

The database's tables (or relations) and their attributes, as well as keys and constraints are presented below:

Note: Only 3 of the relations (Persons, Males, and Females) below have keys, and this is due to the fact that the relations without primary keys allow for any attribute to be a NULL value, since NULL values, while representing a "lack of information" can actually hold more information about a relationship between two individuals and what is unknown. People removed from a family can be signified as a NULL value. If the relations were not allowed to have NULL values and were made by using a composite primary key, it would mean that that row would have to be deleted (by using a cascade) so as to not break the rules of keys (having NULL values in the key values). Essentially, making composite keys in the other relations would force information to be lost if a certain person were to be removed from the Family relation.

Persons

The Persons relation tracks information about individual people. It has the following attributes:

- pID: The primary key for this relation. This value must be unique and non-null for each person.
- full_name: The person's full name.
- gender: The person's gender, which must be either 'M' for male or 'F' for female.
- DoB: The person's date of birth.

Family

The Family relation tracks information about familial relationships between people. It has the following attributes:

- childID: The ID of the child in the family relationship. This value must reference a valid pID in the Persons relation.
- motherID: The ID of the mother in the family relationship. This value must reference a valid pID in the Persons relation, and the person with this ID must have gender 'F'.
- fatherID: The ID of the father in the family relationship. This value must reference a valid pID in the Persons relation, and the person with this ID must have gender 'M'.

Males

The Males relation tracks information about male people. It has the following attributes:

- pID: The primary key for this relation. This value must reference a valid pID in the Persons relation, and the person with this ID must have gender 'M'.

Females

The Females relation tracks information about female people. It has the following attributes:

- pID: The primary key for this relation. This value must reference a valid pID in the Persons relation, and the person with this ID must have gender 'F'.

Brothers

The Brothers relation tracks information about pairs of brothers. It has the following attributes:

- brotherA_ID: The ID of one brother in the pair. This value must reference a valid pID in the Males relation, and the person with this ID must have gender 'M'.

- brotherB_ID: The ID of the other brother in the pair. This value must reference a valid pID in the Males relation, and the person with this ID must have gender 'M'.

The persons associated with brotherA_ID and brotherB_ID must have the same parents.

Sisters

The Sisters relation tracks information about pairs of sisters. It has the following attributes:

- sisterA_ID: The ID of one sister in the pair. This value must reference a valid pID in the Females relation, and the person with this ID must have gender 'F'.
- sisterB_ID: The ID of the other sister in the pair. This value must reference a valid pID in the Females relation, and the person with this ID must have gender 'F'.

The persons associated with sisterA_ID and sisterB_ID must have the same parents.

MaleFemale

The MaleFemale relation tracks information about pairs of people where one is male and the other is female. It has the following attributes:

- maleID: The ID of the male person in the pair. This value must reference a valid pID in the Males relation, and the person with this ID must have gender 'M'.
- femaleID: The ID of the female person in the pair. This value must reference a valid pID in the Females relation, and the person with this ID must have gender 'F'.

Siblings

The Siblings relation tracks information about pairs of siblings, where one is male and the other is female. It has the following attributes:

- brotherID: The ID of the male sibling in the pair. This value must reference a valid pID in the Males relation, and the person with this ID must have gender 'M'.
- sisterID: The ID of the female sibling in the pair. This value must reference a valid pID in the Females relation, and the person with this ID must have gender 'F'.

The persons associated with brotherID and sisterID must have the same parents.

Married

The Married relation tracks information about pairs of married people. It has the following attributes:

- husbandID: The ID of the husband in the pair. This value must reference a valid pID in the Males relation, and the person with this ID must have gender 'M'.
- wifeID: The ID of the wife in the pair. This value must reference a valid pID in the Females relation, and the person with this ID must have gender 'F'.

Give appropriate relational algebra trees-expressions and SQL expressions that return:
a. Children of a given couple;

```
SELECT Mother.full_name AS Mother, Father.full_name AS Father, Child.full_name AS Child
FROM Persons AS Mother, Persons AS Father, Persons AS Child, Family
WHERE Child.pID = Family.childID AND (Mother.pID = (mother ID) AND Father.pID = father ID)) AND
(Mother.pID = Family.motherID AND Father.pID = Family.fatherID);
```

~~~

In the output below, Father ID and Mother ID are 4 and 5, respectively.
~~~

```
SELECT Mother.full_name AS Mother, Father.full_name AS Father, Child.full_name AS Child
FROM Persons AS Mother, Persons AS Father, Persons AS Child, Family
WHERE Child.pID = Family.childID AND (Mother.pID = 5 AND Father.pID = 4) AND
(Mother.pID = Family.motherID AND Father.pID = Family.fatherID);

# Father ID and Mother ID are 4 and 5, respectively
```

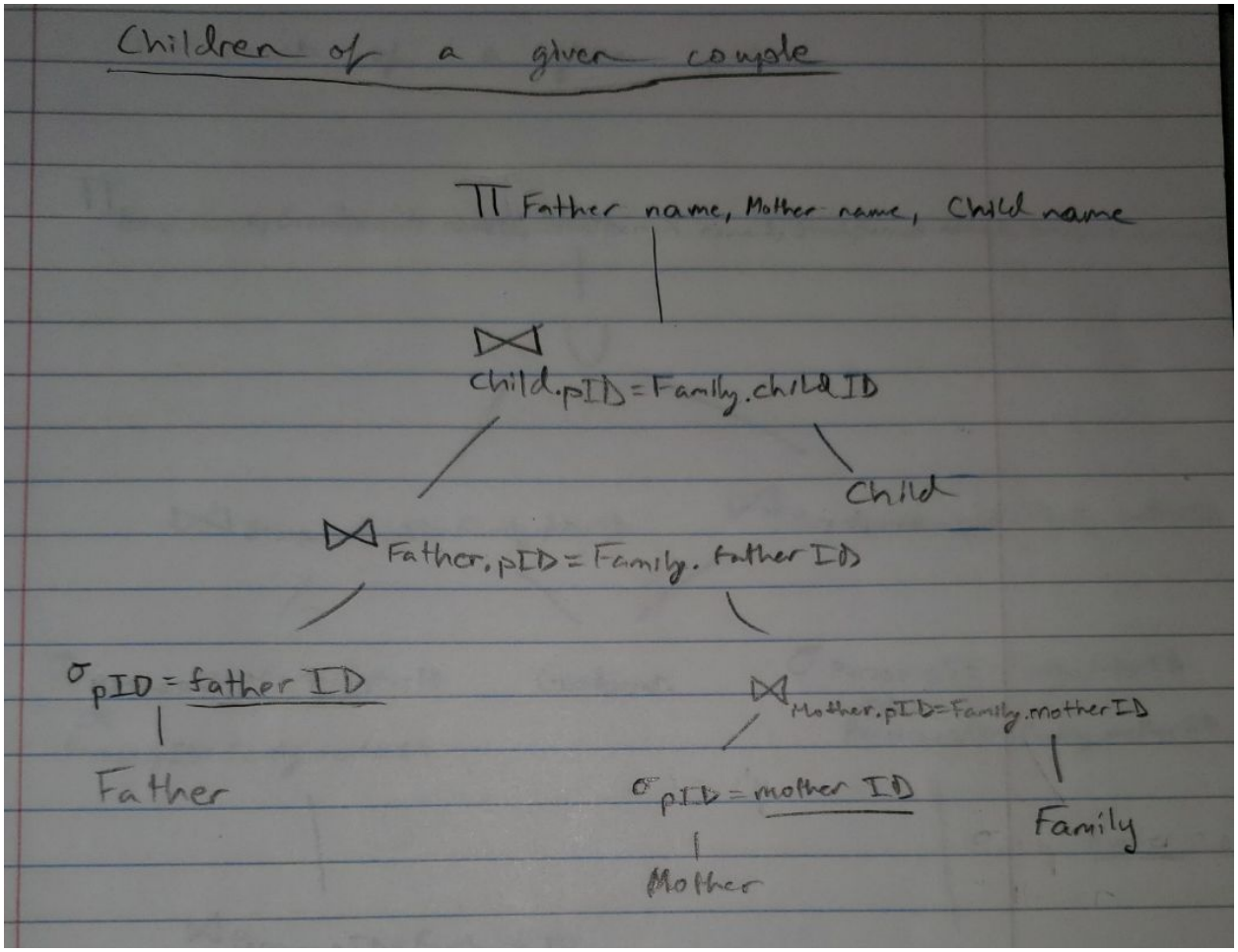
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Mother	Father	Child
Mother	Father	Me
Mother	Father	Sister
Mother	Father	Brother

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Mother	Father	Child
Mother	Father	Me
Mother	Father	Sister
Mother	Father	Brother

~~~

Note: Please note that this query is made so that the highlighted portions of the code are to be replaced with whatever the user wishes said values to be. In the relational algebra trees, these values are represented by them being underlined.
~~~



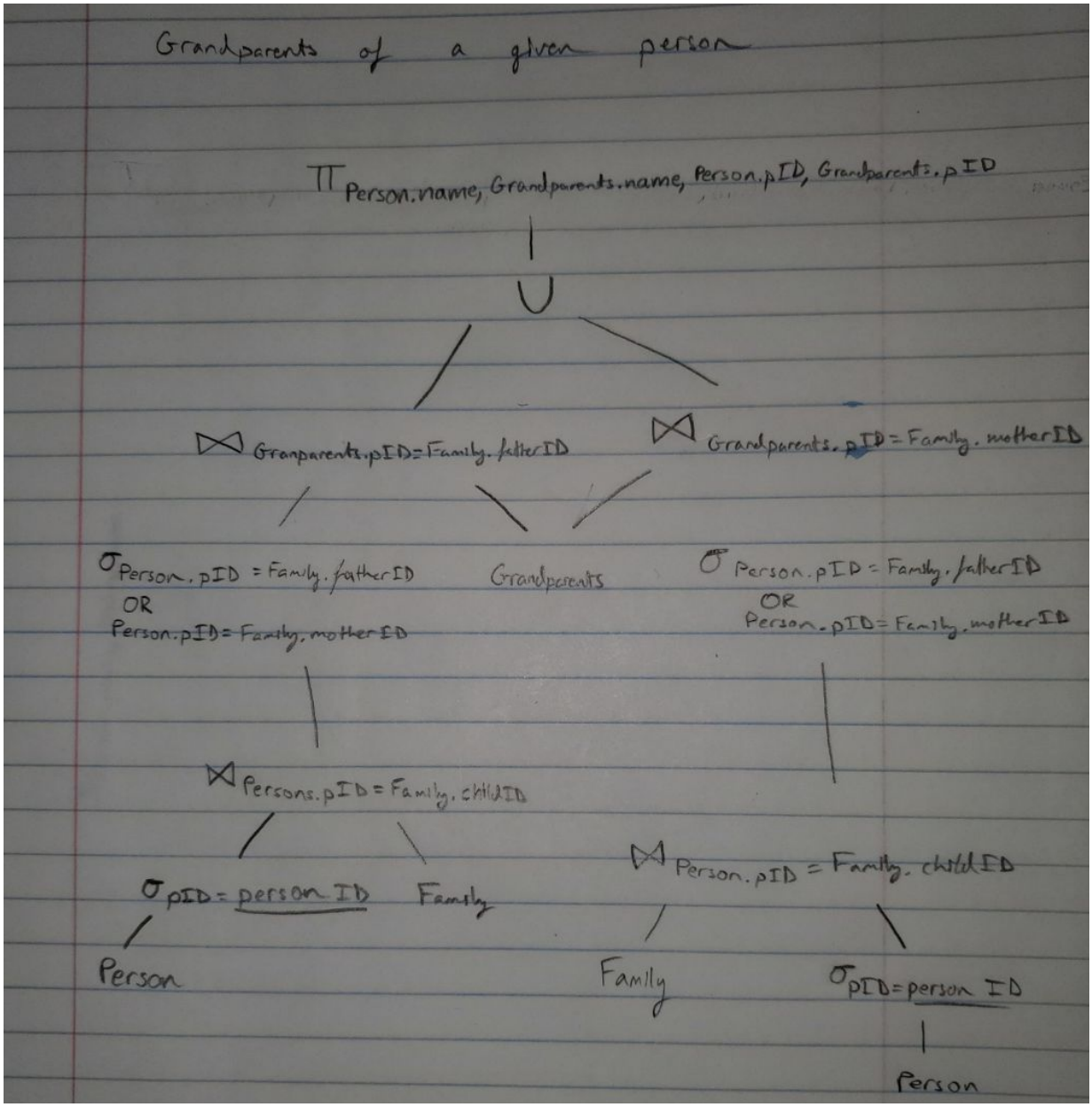
b. Grandparents of a given person

```
# Person is a mother and their child is also a parent
SELECT Person.full_name AS Person, Grandparents.full_name AS Grandparents
FROM Persons AS Person, Persons AS Grandparents, Family
WHERE Person.pID = (person ID) AND Grandparents.pID = Family.motherID AND
Family.childID IN
        (SELECT Persons.pID
        FROM Persons, Family
        WHERE Family.childID = (person ID) AND (Persons.pID = Family.motherID OR
        Persons.pID = Family.fatherID))
UNION
# Person is a father and their child is also a parent
SELECT Person.full_name AS Person, Grandparents.full_name AS Grandparents
FROM Persons AS Person, Persons AS Grandparents, Family
```

WHERE Person.pID = (person ID) AND Grandparents.pID = Family.fatherID AND Family.childID IN
(SELECT Persons.pID
FROM Persons, Family
WHERE Family.childID = (person ID) AND (Persons.pID = Family.motherID OR Persons.pID = Family.fatherID));

~~~

Person ID is 1 in the query used to test.

```
12  SELECT Person.full_name AS Person, Grandparents.full_name AS Grandparents
13  FROM Persons AS Person, Persons AS Grandparents, Family
14  WHERE Person.pID = (1) AND Grandparents.pID = Family.motherID AND Family.childID IN
15      (SELECT Persons.pID
16      FROM Persons, Family
17  WHERE Family.childID = (1) AND (Persons.pID = Family.motherID OR Persons.pID = Family.fatherID))
18  UNION
19  # Person is a father and their child is also a parent
20  SELECT Person.full_name AS Person, Grandparents.full_name AS Grandparents
21  FROM Persons AS Person, Persons AS Grandparents, Family
22  WHERE Person.pID = (1) AND Grandparents.pID = Family.fatherID AND Family.childID IN
23      (SELECT Persons.pID
24      FROM Persons, Family
25  WHERE Family.childID = (1) AND (Persons.pID = Family.motherID OR Persons.pID = Family.fatherID));
26
27  # Person ID is 1
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	Grandparents
Me	Grandmother (paternal)
Me	Grandmother (maternal)
Me	Grandfather (paternal)
Me	Grandfather (maternal)

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	Grandparents
Me	Grandmother (paternal)
Me	Grandmother (maternal)
Me	Grandfather (paternal)
Me	Grandfather (maternal)

~~~

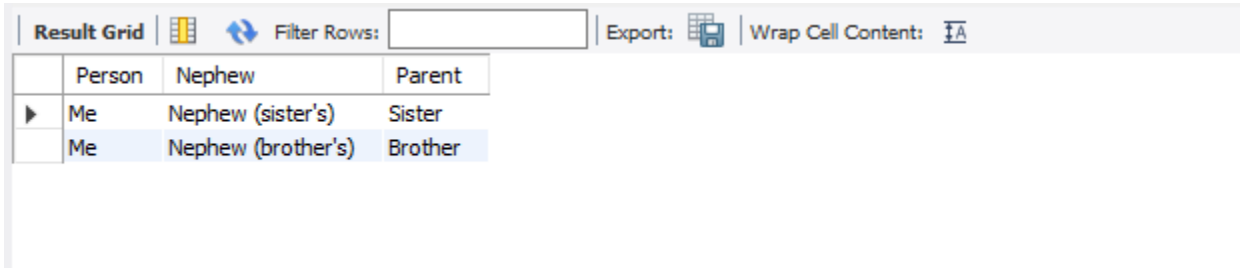
Note: Please note that this query is made so that the highlighted portions of the code are to be replaced with whatever the user wishes said values to be. In the relational algebra trees, these values are represented by them being underlined.



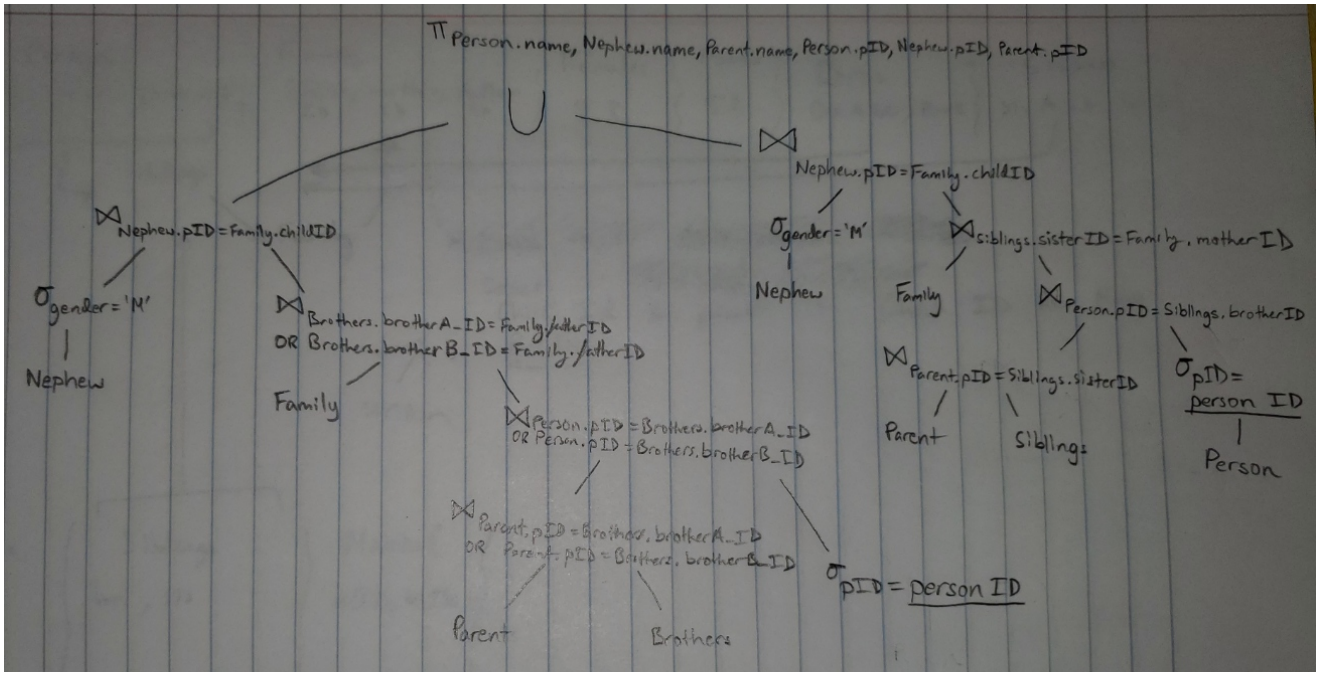
c. Nephews -- sons of one's brother or sister -- of a given person

```
SELECT Person.full_name AS Person, Nephew.full_name AS Nephew, Parent.full_name AS
Parent
FROM Persons AS Person, Persons AS Nephew, Persons AS Parent, Family
WHERE Person.pID = (person ID) AND Nephew.pID = Family.childID AND Nephew.gender =
'M' AND Parent.pID = Family.motherID AND Family.motherID IN
        (SELECT Persons.pID
        FROM Persons, Siblings
        WHERE Siblings.brotherID = (person ID) AND Persons.pID = Siblings.sisterID)
```

```
UNION
SELECT Person.full_name AS Person, Nephew.full_name AS Nephew, Parent.full_name AS
Parent
FROM Persons AS Person, Persons AS Nephew, Persons AS Parent, Family
WHERE Person.pID = (person ID) AND Nephew.pID = Family.childID AND Nephew.gender =
'M' AND Parent.pID = Family.fatherID AND Family.fatherID IN
        (SELECT Persons.pID
        FROM Persons, Brothers
        WHERE (brotherA_ID = (person ID) OR brotherB_ID = (person ID)) AND
        (brotherA_ID = Persons.pID OR brotherB_ID = Persons.pID));
```

~~~

Person ID is 1 in the query used to test.

```
29    ## Nephews -- sons of one's brother or sister -- of a given person
30
31    SELECT Person.full_name AS Person, Nephew.full_name AS Nephew, Parent.full_name AS Parent
32    FROM Persons AS Person, Persons AS Nephew, Persons AS Parent, Family
33    WHERE Person.pID = (1) AND Nephew.pID = Family.childID AND Nephew.gender = 'M' AND Parent.pID = Family.motherID AND Family.motherID IN
34        (SELECT Persons.pID
35            FROM Persons, Siblings
36            WHERE Siblings.brotherID = (1) AND Persons.pID = Siblings.sisterID)
37    UNION
38    SELECT Person.full_name AS Person, Nephew.full_name AS Nephew, Parent.full_name AS Parent
39    FROM Persons AS Person, Persons AS Nephew, Persons AS Parent, Family
40    WHERE Person.pID = (1) AND Nephew.pID = Family.childID AND Nephew.gender = 'M' AND Parent.pID = Family.fatherID AND Family.fatherID IN
41        (SELECT Persons.pID
42            FROM Persons, Brothers
43    WHERE (brotherA_ID = (1) OR brotherB_ID = (1)) AND (brotherA_ID = Persons.pID OR brotherB_ID = Persons.pID));
44
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	Nephew	Parent
Me	Nephew (sister's)	Sister
Me	Nephew (brother's)	Brother



~~~

Note: Please note that this query is made so that the highlighted portions of the code are to be replaced with whatever the user wishes said values to be. In the relational algebra trees, these values are represented by them being underlined.


π Person.name, Nephew.name, Parent.name, Person.pID, Nephew.pID, Parent.pID
∪
⋈ Nephew.pID = Family.childID
σ gender = 'M'
Nephew
⋈ Brothers.brotherA_ID = Family.fatherID OR Brothers.brotherB_ID = Family.fatherID
Family
⋈ Person.pID = Brothers.brotherA_ID OR Person.pID = Brothers.brotherB_ID
⋈ Parent.pID = Brothers.brotherA_ID OR Parent.pID = Brothers.brotherB_ID
Parent
Brothers
σ pID = person ID
⋈ Nephew.pID = Family.childID
σ gender = 'M'
Nephew
⋈ Siblings.sisterID = Family.motherID
Family
⋈ Person.pID = Siblings.brotherID
⋈ Parent.pID = Siblings.SisterID
Parent
Siblings
σ pID = person ID
Person

Part B

Question:

The definition of a brother-in-law in the Cambridge English Dictionary is:
a. The husband of a person's sister,
b. The brother of a person's wife or husband, or
c. The husband of the sister of a person's wife or husband

Given the relation Brothers that has tuples of the form (c, d), where c is the brother of d, the relation Sisters that consists of tuples of the form (g, h), where g is the sister of h, the relation Brother-Sister which has tuples of the form (e, f), where e is the brother and f is the sister, and the relation Husband-Wife that has tuples of the form (a, b), where a is the husband and b is the wife:

1. Describe how you would define the relation Brother-in-Law whose tuples have the form (x, y) with x being the brother-in-law of y.

Part B

(a, b) : (Husband, Wife)
(c, d) : (Brother A, Brother B)
(e, f) : (Brother, Sister)
(g, h) : (Sister A, Sister B)

a) $x = a$ where $b = f|_{y=e}$, if y is male, or
$x = a$ where $b = g|_{y=h}$ or $b = h|_{y=g}$, if y is female
Person's brother-in-law is a husband, the wife is also a sister, and the person used to specify is their sibling.

b) $x = e$ where $f = b|_{y=e}$, if y is male, or
$x = c$ where $d = a|_{y=b}$, or
$x = d$ where $c = a|_{y=b}$, if y is female
The person's brother-in-law is also a brother, and their sibling (the brothers) is married to the person.

[illegible]

c) $x = a$ where $(b = g$ and $h = b|_{y=a})$ or
$(b = h$ and $g = b|_{y=a})$,
if y is male, or
$x = a$ where $b = f$ and $e = a|_{y=b}$, if y is female
The person's BiL is a husband, and their wife is a sister/sibling to the one who is married to the person.

2. Give appropriate relational algebra-tree and SQL expressions that produce the relation brother-in-Law.

```
# case a
SELECT Person.full_name AS Person, BrotherInLaw.full_name as BrotherInLaw
FROM Persons AS Person, Persons AS BrotherInLaw, Married
WHERE Person.pID = (person ID) AND BrotherInLaw.pID = husbandID AND wifeID IN
        (SELECT sisterID
   FROM Siblings
   WHERE brotherID = (person ID))
UNION
# case b
SELECT Person.full_name AS Person, BrotherInLaw.full_name as BrotherInLaw
FROM Persons AS Person, Persons AS BrotherInLaw, Siblings
WHERE person.pID = (person ID) AND BrotherInLaw.pID = Siblings.brotherID AND
Siblings.sisterID IN
        (SELECT wifeID
   FROM Married
   WHERE husbandID = (person ID))
UNION
# case c
SELECT Person.full_name AS Person, BrotherInLaw.full_name as BrotherInLaw
FROM Persons AS Person, Persons AS BrotherInLaw, Married
WHERE person.pID = (person ID) AND BrotherInLaw.pID = husbandID AND wifeID IN
        (SELECT sisterA_ID
   FROM Sisters, Married
   WHERE sisterB_ID = wifeID AND husbandID = (person ID)
   UNION
   SELECT sisterB_ID
   FROM Sisters, Married
   WHERE sisterA_ID = wifeID AND husbandID = (person ID));
~~~
```

Person ID is 1 in the query used to test.

```sql
## Brother-In-Law

# case a
SELECT Person.full_name AS Person, BrotherInLaw.full_name as BrotherInLaw
FROM Persons AS Person, Persons AS BrotherInLaw, Married
WHERE Person.pID = (1) AND BrotherInLaw.pID = husbandID AND wifeID IN
    (SELECT sisterID
    FROM Siblings
    WHERE brotherID = (1))
UNION
# case b
SELECT Person.full_name AS Person, BrotherInLaw.full_name as BrotherInLaw
FROM Persons AS Person, Persons AS BrotherInLaw, Siblings
WHERE person.pID = (1) AND BrotherInLaw.pID = Siblings.brotherID AND Siblings.sisterID IN
    (SELECT wifeID
    FROM Married
```

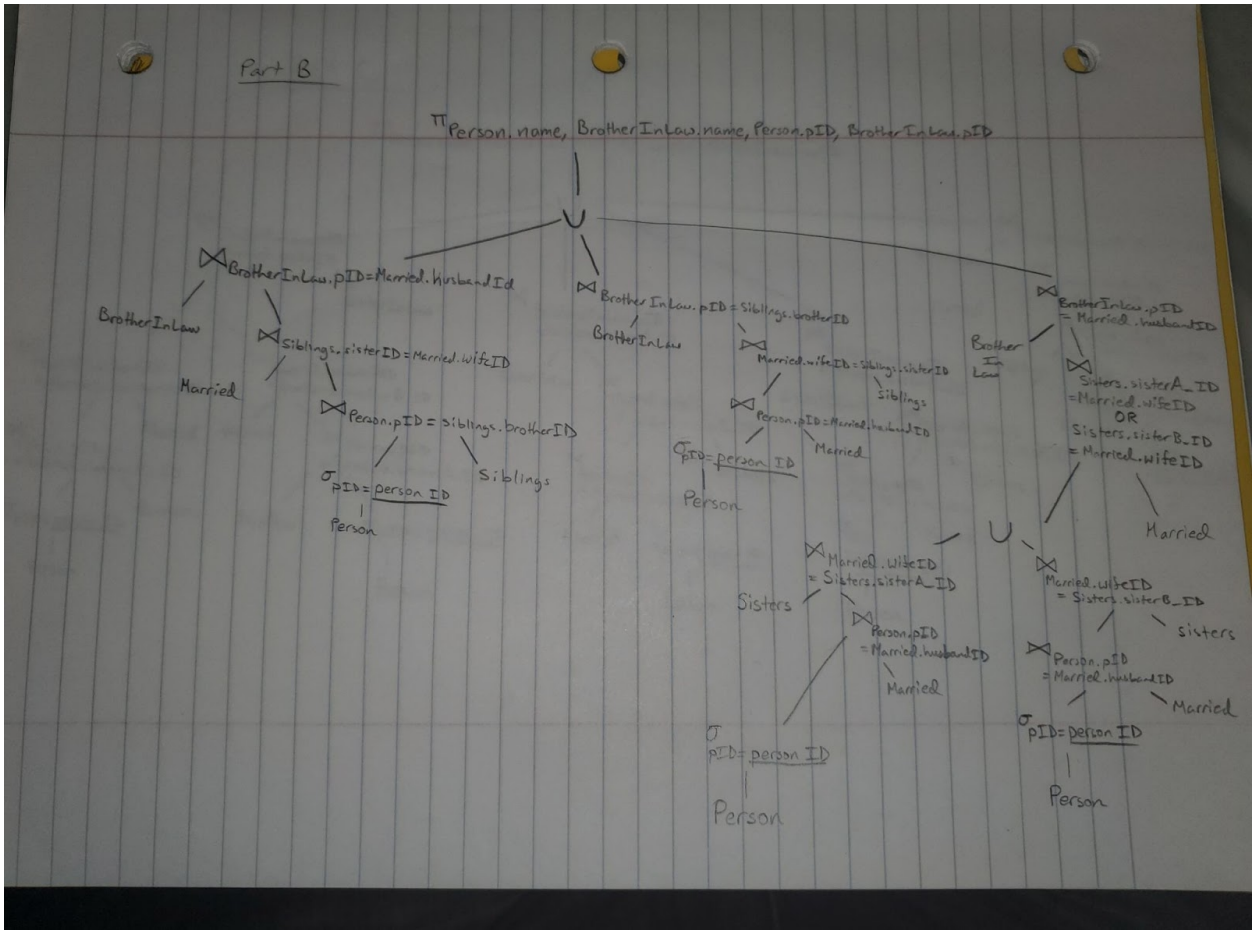
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	BrotherInLaw
Me	Sister's husband (BIL a)
Me	Wife's brother (BIL b)
Me	Wife's sister's husband (BIL c)

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	BrotherInLaw
Me	Sister's husband (BIL a)
Me	Wife's brother (BIL b)
Me	Wife's sister's husband (BIL c)

~~~

Note: Please note that this query is made so that the highlighted portions of the code are to be replaced with whatever the user wishes said values to be. In the relational algebra trees, these values are represented by them being underlined.
~~~



Part C

Question:

The definition of a sister-in-law in the Cambridge English Dictionary is:
a. The wife of a person's brother, b. The sister of a person's wife or husband, or c. The wife of the brother of a person's wife or husband

Given the relations Brothers, Sisters, Brother-Sister, and Husband-Wife as in B above:

1. Describe how you would define the relation Sister-in-Law whose tuples have the form (x, y) with x being the sister-in-law of y.

Part C

a) $x=b$ where $a=c|_{y=d}$ or $a=d|_{y=c}$, if y is male or,

$x=b$ where $a=e|_{y=f}$, if y is female

The person's sister-in-law is a wife, and their husband is also a brother and the person is their sibling

b) $x=g$ where $h=b|_{y=a}$ or $x=h$ where $g=b|_{y=a}$, if y is male

or

$x=f$ where $e=a|_{y=b}$ if y is female

The person's sister-in-law is a sister, and the sibling is also married to the given person

c) $x=b$ where $a=e$ and $f=b|_{y=a}$, if y is male, or

$x=b$ where $(a=c$ and $d=a|_{y=b})$ or $(a=d$ and $c=a|_{y=b})$, if y is female.

The person's sister-in-law is a wife, and their wife is siblings with another individual that is married to the given person.

2. Give appropriate relational algebra-tree and SQL expressions that produce the relation Sister-in-Law.

```
# case A
SELECT Person.full_name AS Person, SisterInLaw.full_name as SisterInLaw
FROM Persons AS Person, Persons AS SisterInLaw, Married
WHERE Person.pID = (person ID) AND SisterInLaw.pID = wifeID AND husbandID IN
        (SELECT brotherA_ID
        FROM Brothers
        WHERE brotherB_ID = (person ID)
        UNION
        SELECT brotherB_ID
        FROM Brothers
        WHERE brotherA_ID = (person ID))
UNION
# case B
SELECT Person.full_name AS Person, SisterInLaw.full_name as SisterInLaw
FROM Persons AS Person, Persons AS SisterInLaw, Siblings
WHERE Person.pID = (person ID) AND SisterInLaw.pID IN
        (SELECT sisterA_ID
        FROM Sisters, Married
        WHERE sisterB_ID = wifeID AND husbandID = (person ID)
        UNION
        SELECT sisterB_ID
        FROM Sisters, Married
        WHERE sisterA_ID = wifeID AND husbandID = (person ID))
UNION
# case C
SELECT Person.full_name AS Person, SisterInLaw.full_name as SisterInLaw
FROM Persons AS Person, Persons AS SisterInLaw, Married
WHERE person.pID = (person ID) AND SisterInLaw.pID = wifeID AND husbandID IN
        (SELECT brotherID
        FROM Siblings, Married
        WHERE sisterID = wifeID AND husbandID = (person ID));
```

~~~

Person ID is 1 in the query used to test.
~~~

```
## Sister-In-Law

# case A
SELECT Person.full_name AS Person, SisterInLaw.full_name as SisterInLaw
FROM Persons AS Person, Persons AS SisterInLaw, Married
WHERE Person.pID = (1) AND SisterInLaw.pID = wifeID AND husbandID IN
    (SELECT brotherA_ID
        FROM Brothers
        WHERE brotherB_ID = (1)
        UNION
        SELECT brotherB_ID
        FROM Brothers
        WHERE brotherA_ID = (1))
UNION
# case B
SELECT Person.full_name AS Person, SisterInLaw.full_name as SisterInLaw
```

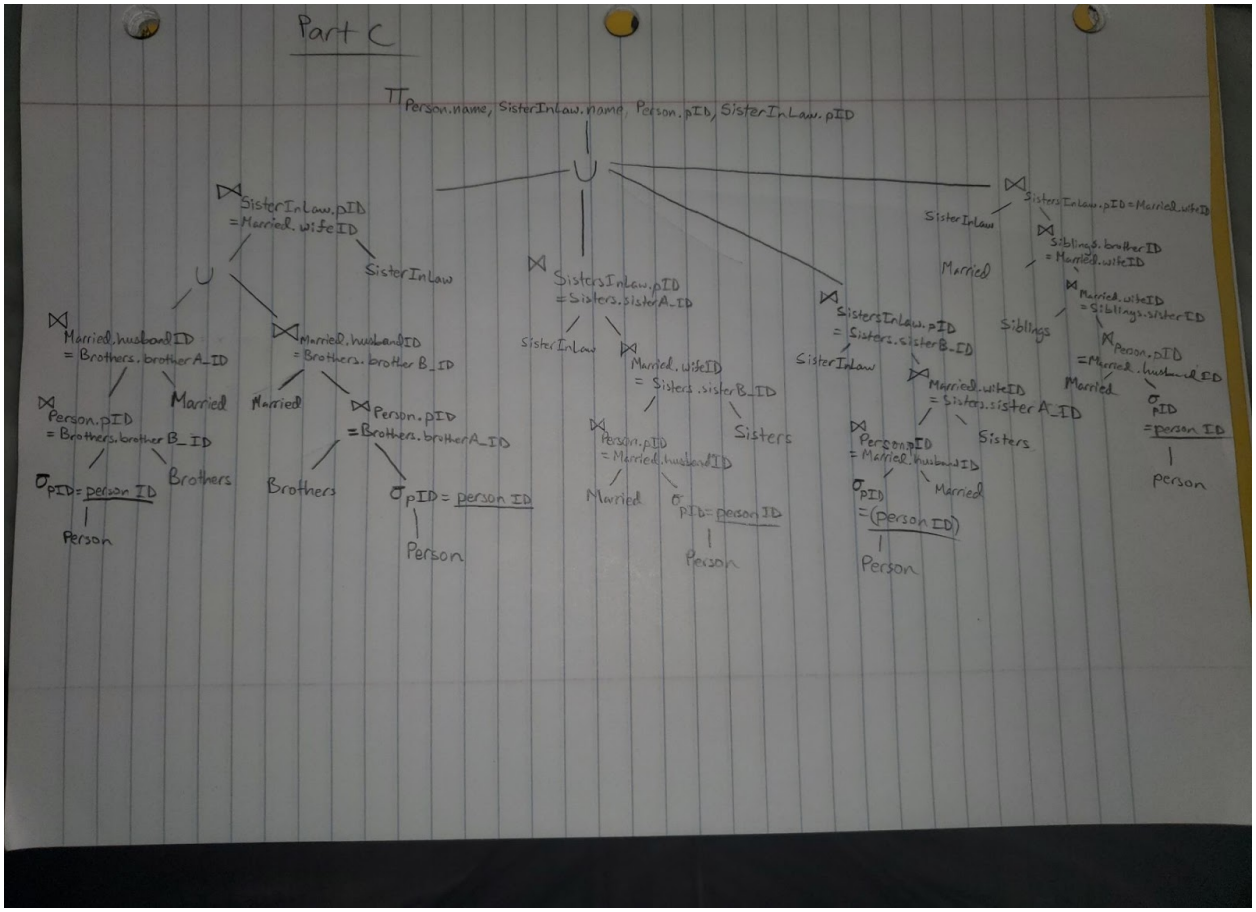
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	SisterInLaw
Me	Brother's wife (SIL a)
Me	Wife's sister (SIL b)
Me	Wife's brother's wife (SIL c)

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Person	SisterInLaw
Me	Brother's wife (SIL a)
Me	Wife's sister (SIL b)
Me	Wife's brother's wife (SIL c)

~~~

Note: Please note that this query is made so that the highlighted portions of the code are to be replaced with whatever the user wishes said values to be. In the relational algebra trees, these values are represented by them being underlined.
~~~



Part D

Question:

Recursively, build up a family tree given a person in the family.

```
WITH RECURSIVE FamilyTree AS(

# Initialization
   SELECT P.pID, P.full_name, F.fatherID, F.motherID
   FROM Persons P JOIN Family F on P.pID = F.childID
   WHERE P.pID = (person ID)

        UNION ALL

# Recursion step
        SELECT P.pID, P.full_name,
        (SELECT F.fatherID FROM Family F WHERE F.childID = P.pID),
        (SELECT F.motherID FROM Family F WHERE F.childID = P.pID)
```

```
    FROM Persons P JOIN FamilyTree T ON P.pID = T.fatherID

    UNION ALL

# Recursion step
        SELECT P.pID, P.full_name,
        (SELECT F.fatherID FROM Family F WHERE F.childID = P.pID),
        (SELECT F.motherID FROM Family F WHERE F.childID = P.pID)

    FROM Persons P JOIN FamilyTree T ON P.pID = T.motherID)

SELECT F.pID, F.full_name,
(SELECT P.full_name FROM Persons P WHERE P.pID = F.fatherID) AS "Father",
(SELECT P.full_name FROM Persons P WHERE P.pID = F.motherID) AS "Mother"

FROM FamilyTree F;
```

Explanation:

This recursive query identifies the selected person by their ID (noted by the highlighted part of the code). This beginning part of the code (marked as *Initialization*) also takes the mother and father IDs of the person ID provided to the query by looking through the Persons and Family relations.

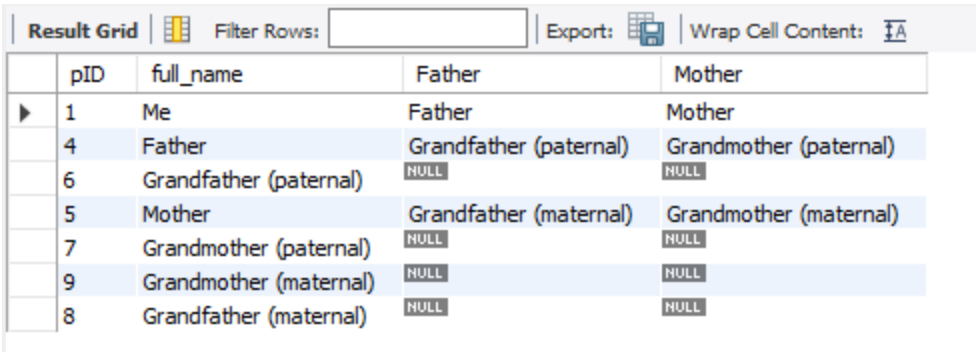
Then, the query has two recursion steps, which are combined by using the UNION ALL operator to combine the results of the recursive steps. Both steps select a person's ID, full name, and their parents' IDs. This part of the query is what allows the entire family tree to be made after the initialization step.

Lastly, for the output of the query, the query outputs (or projects - if using relational algebra nomenclature) each person's ID, full name, and the full names of their parents (which of

course can also be NULL, should their parents be absent in the tables/relations of the database), which would conclude this query.

Please note that in the provided screenshot of the code I used person ID as being 1 as that would make a full family tree in the case of the data I had in my database.

~~~

Person ID is 1 in the query used to test.

```
1  WITH RECURSIVE FamilyTree AS(
2
3  # Initialization
4      SELECT P.pID, P.full_name, F.fatherID, F.motherID
5      FROM Persons P JOIN Family F on P.pID = F.childID
6      WHERE P.pID = (1)
7
8      UNION ALL
9
10 # Recursion step
11         SELECT P.pID, P.full_name,
12         (SELECT F.fatherID FROM Family F WHERE F.childID = P.pID),
13         (SELECT F.motherID FROM Family F WHERE F.childID = P.pID)
14
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

pID	full_name	Father	Mother
1	Me	Father	Mother
4	Father	Grandfather (paternal)	Grandmother (paternal)
6	Grandfather (paternal)	NULL	NULL
5	Mother	Grandfather (maternal)	Grandmother (maternal)
7	Grandmother (paternal)	NULL	NULL
9	Grandmother (maternal)	NULL	NULL
8	Grandfather (maternal)	NULL	NULL
~~~



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

pID	full_name	Father	Mother
1	Me	Father	Mother
4	Father	Grandfather (paternal)	Grandmother (paternal)
6	Grandfather (paternal)	NULL	NULL
5	Mother	Grandfather (maternal)	Grandmother (maternal)
7	Grandmother (paternal)	NULL	NULL
9	Grandmother (maternal)	NULL	NULL
8	Grandfather (maternal)	NULL	NULL

Part E

Question:

Implement A through D above using your RDBS of choice. Test your implementation of the queries using an appropriate set of data. The data utilized must be representative and sufficient to demonstrate the validity of your queries.

Please refer throughout the document to see screenshots corresponding with each part (the output of the queries is shown).

Part F

Question:

Write a CONSTRAINT statement that checks that the brother-in-law and sister-in-law of a given person must not be a brother and sister. Implement the CONSTRAINT using a function and a stored procedure. Show how a TRANSACTION may be used to incorporate the CONSTRAINT, specifying the appropriate type and isolation level of the TRANSACTION. Discuss the atomicity problems, if any, that could occur should the system crash between the two updates.

Upon trying to complete this part of the project, I was unable to produce a CONSTRAINT statement that would check if a brother-in-law and sister-in-law of a given person must not be a brother and sister (in the Siblings relation). Creation of the CONSTRAINT is not possible provided the constraints of MySQL. Perhaps it is possible in a different version of SQL.

Regardless, making the transaction also seemed to not work for me (despite trying out multiple different queries). Instead I will say what I would have done.

I would've made a transaction which would have the isolation level SERIALIZABLE. Doing this would make sure that the transaction of the problem runs in the order in which it was started in the code.

The transaction should then have SAVEPOINT statements that would allow the transaction to reverse itself should the violation of a brother-in-law and sister-in-law of a given brother be siblings. We could use the code for parts B and C and make some function(s) out of them that checks for the brother-in-law and sister-in-law (collects the information provided the person ID), and then checks the siblings table. If the violation were to be found (like using conditional statements), the transaction would rollback, essentially acting as a constraint.

That would be my plan for coding this part up. Please note the reason for choosing the SERIALIZABLE isolation level would be because it prevents dirty, phantom, and non-repeatable reads. While this problem would likely not arise in the context of the project I had made (where the idea would be to just run the function once at a time already, if the database were to actually be used for a more used reason, and so more transactions would probably be running at a time, this would definitely help the database system be more stable and not have issues with the data in the family schema.

Part G

Question: The data utilized, whether your own personal data or available elsewhere, must be representative and sufficient to demonstrate the validity of all outputs.

The data was representative enough to show off the parts of the project I was able to complete.