5. Sélectionnez tous les professeurs et leur âge.

   Vous avez normalement remarqué que la liste est incomplète. En effet, si nous demandons l'âge de tous les professeurs, cela induit que tous les professeurs possèdent la propiété `prop:age` *, or ce n'est pas le cas. Seul Edouard apparaît dans la liste.*

```
9  select ?teacher ?age where {
10   ?teacher a job:teacher ;
11         prop:age ?age .
12 }
13
14
```

Graph | XML/RDF | Table | Validate

num	?teacher	?age
1	<http://uca.fr/family/member/Edouard>	40

6. Sélectionnez tous les parents présent dans la base.

   Ici, on remarque que des individus apparaîssent plusieurs fois dans les résultats. En effet, plusieurs enfants ont les mêmes parents : Yannick et Victoria. Une solution serait de filtrer les doublons tel que : `select distinct [...]`.

   Appliquer la modification et vérifier qu'aucun doublon ne soit présent.

```
9  select distinct ?parent where {
10   ?child prop:parent ?parent .
11 }
12
13
```

Graph | XML/RDF | Table | Validate

num	?parent
1	<http://uca.fr/family/member/Jean>
2	<http://uca.fr/family/member/Caroline>
3	<http://uca.fr/family/member/Edouard>
4	<http://uca.fr/family/member/Jessica>

7. Sélectionnez tous les individus de la base ainsi que leur taille, dont celle-ci est inférieure ou égale à 1m70.

   La fonction `filter([...])` *est très utile dans ce type de requête.*

```
9  select ?individual ?height where {
10   ?individual prop:size ?height .
11   FILTER (?height <= "170"^^<http://www.w3.org/2001/XMLSchema#decimal>)
12 }
13
14
15
```

Graph | XML/RDF | Table | Validate

num	?individual	?height
1	<http://uca.fr/family/member/Jessica>	"170.0"^^xsd:decimal
2	<http://uca.fr/family/member/Jean>	"160.0"^^xsd:decimal
3	<http://uca.fr/family/member/Caroline>	"160.0"^^xsd:decimal

8. À l'aide de la requête `construct` (dont le template est disponible dans *Query → Construct*), élaborez une requête pour visualiser les liens de parenté (`prop:parent`) entre les individus.

```
CONSTRUCT {
  ?child prop:parent ?parent .
}
WHERE {
  ?child prop:parent ?parent .
}
```

Graph | XML/RDF | Table | Validate

num	?child	?parent
1	<http://uca.fr/family/member/Edouard>	<http://uca.fr/family/member/Jean>
2	<http://uca.fr/family/member/Edouard>	<http://uca.fr/family/member/Caroline>
3	<http://uca.fr/family/member/Yannick>	<http://uca.fr/family/member/Edouard>
4	<http://uca.fr/family/member/Yannick>	<http://uca.fr/family/member/Jessica>
5	<http://uca.fr/family/member/Victoria>	<http://uca.fr/family/member/Edouard>
6	<http://uca.fr/family/member/Victoria>	<http://uca.fr/family/member/Jessica>

9. A l'aide de la requête `ask` (dont le template est disponible dans *Query → Ask*), déterminez si Jean est bien le grand-père de Victoria.
Le résultat s'affiche sous le format XML et la réponse se formule avec un booléen : `true` *ou* `false`.

```
ASK
WHERE {
  ?jean prop:parent ?edouard .
  ?edouard prop:parent ?victoria .
}
```

Graph | XML/RDF | Table | Validate

```
<?xml version="1.0" ?>
<sparql xmlns='http://www.w3.org/2005/sparql-results#'>
<head>
</head>
<boolean>true</boolean>
</sparql>
```

Three examples of queries I found:

Select all children and their ages:

```
SELECT ?child ?age
WHERE {
   ?child a type:Person ;
         prop:age ?age .
   FILTER (?age < 18)  # Optionally, filter only minors
}
```

Graph | XML/RDF | Table | Validate

num	?child	?age
1	<http://uca.fr/family/member/Yannick>	17
2	<http://uca.fr/family/member/Victoria>	10

Select all individuals who are not teachers:

```
SELECT ?individual
WHERE {
   ?individual a type:Person ;
            prop:gender ?gender .
   FILTER NOT EXISTS { ?individual a job:teacher }
}
```

Graph | XML/RDF | Table | Validate

num	?individual
1	<http://uca.fr/family/member/Jessica>
2	<http://uca.fr/family/member/Jean>
3	<http://uca.fr/family/member/Caroline>
4	<http://uca.fr/family/member/Yannick>
5	<http://uca.fr/family/member/Victoria>

Select the names and sizes of all individuals:

```
SELECT ?name ?size
WHERE {
   ?individual a type:Person ;
               prop:name ?name ;
               prop:size ?size .
}
```

Graph | XML/RDF | Table | Validate

num	?name	?size
1	Edouard	"180.0"^^xsd:decimal
2	Jessica	"170.0"^^xsd:decimal
3	Jean	"160.0"^^xsd:decimal
4	Caroline	"160.0"^^xsd:decimal