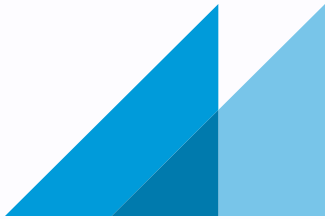
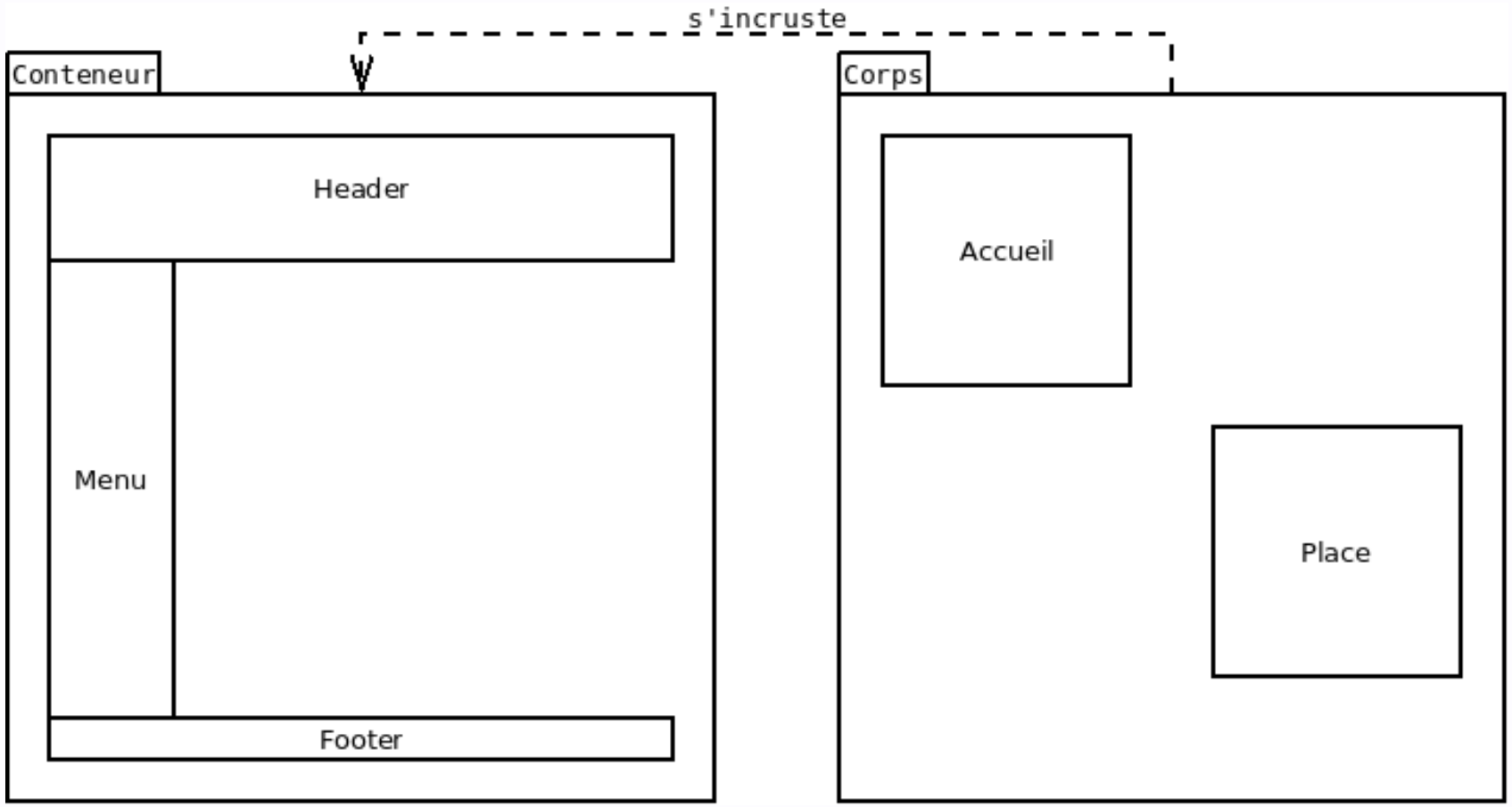
Penser les données par un ORM: les modèles

Maxime Challon -- EnC -- Atos



Héritages de templates

Extension de templates



Extends: le conteneur

```
<!-- partials/conteneur.html -->
<!doctype html>
<html lang="en">
<head>
    <title>Factbook | {{sous_titre}}</title>
</head>
<body>
    {% block body%}{%endblock%}
</body>

</html>
```

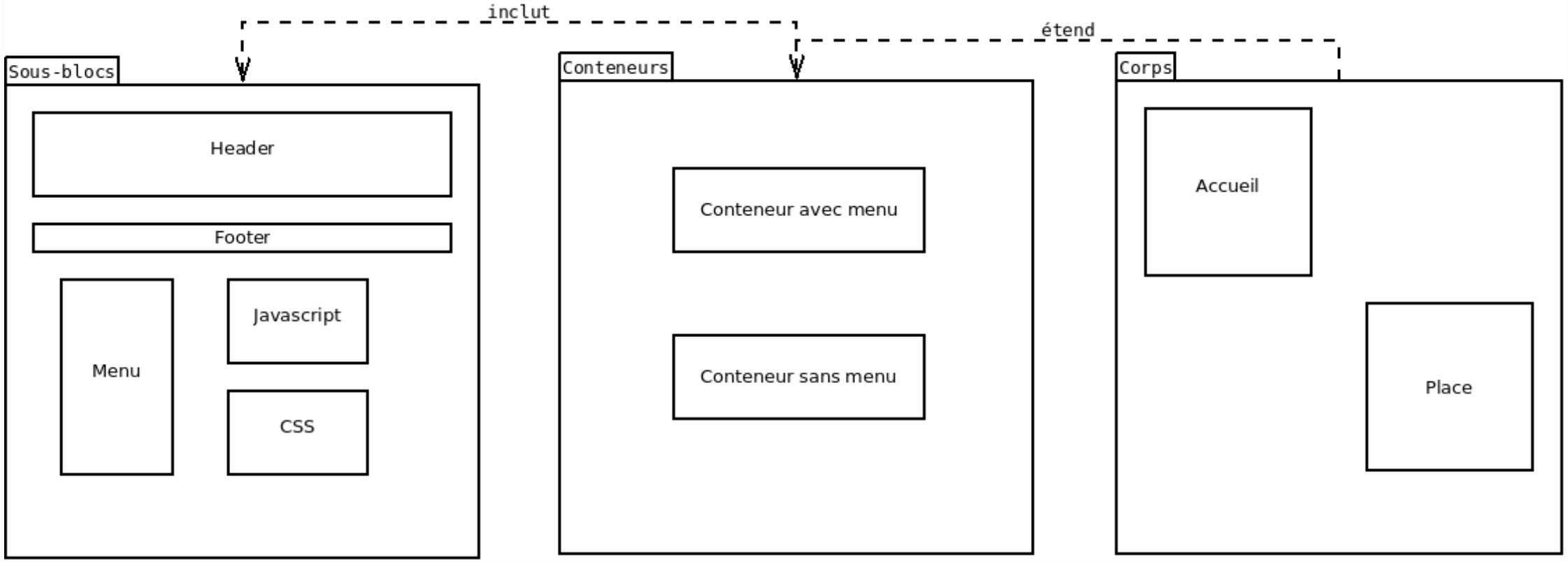
Extends: le contenu

```
<!-- pages/pays.html -->
{% extends "partials/conteneur.html" %}

{% block body %}

    <div class="container">
        <h1>Bienvenue sur l'application du Factbook!</h1>
        <p>Voici le lien vers le site officiel : <a
                href="https://www.cia.gov/the-world-factbook/">
                https://www.cia.gov/the-world-factbook/</a></p>
    </div>
{% endblock %}
```

Un petit récapitulatif de la structure de l'application

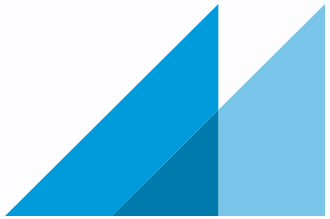
```
app
| app.py
| templates/
|  |-- partials/
|  |  |-- conteneur.html
|  |-- pages/
|  |  |-- pays.html
| routes/
|  |-- __init__.py
|  |-- generales.py
```

Inclusion de templates



Inclusion

```
<!-- partials/conteneur.html -->
<head>
    <meta charset="UTF-8">
    <title>{%block titre %}{%endblock%} | {{sous_titre}}</title>
    {% include "partials/css.html" %}
    {% include "partials/metadata.html" %}
    {% include "partials/js.html" %}
    {% block js %}{%endblock%}
    {% block css %}{%endblock%}
</head>
```



Liens internes

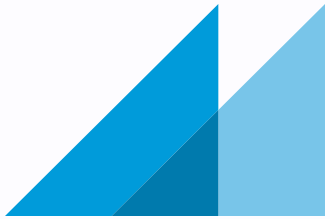
Liens internes

A ne pas faire !

```
<a href="localhost:5000/pays/{{nom_pays}}">{{nom_pays}}</a>
```

Utiliser `url_for()`

```
<a href="{{url_for('pays', nom=nom_pays)}}">{{nom_pays}}</a>
```



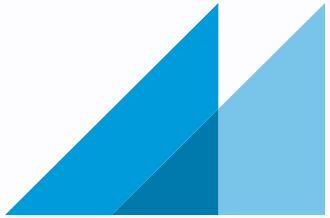
Statics

Rangement

```
| app
|   |-- static
|   |   |-- css
|   |   |   |-- css.css
|   |   |-- js
|   |   |   |-- js.js
|   |   |-- img
```

Appels

```
<link rel="stylesheet" href="{{ url_for('static',
filename='css/css.css')}}">
```



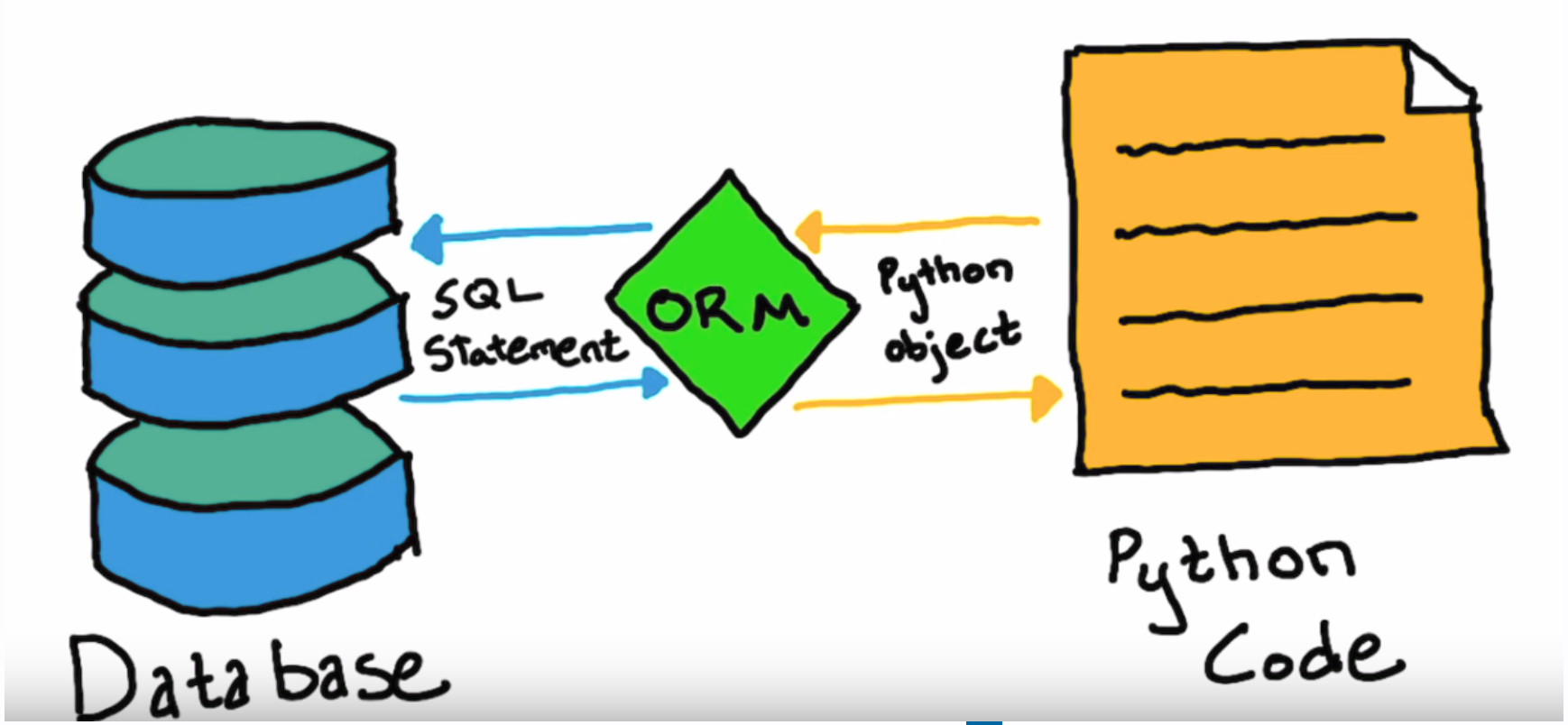
Penser les données par un ORM: les modèles

Bienvenue sur l'application du Factbook!

Voici le lien vers le site officiel : https://www.cia.gov/the-world-factbook/

#	Nom	Capitale	Continent
1	Algeria	inconnu	inconnu
2	Angola	inconnu	inconnu
3	Botswana	inconnu	inconnu
4	Benin	inconnu	inconnu
5	Burundi	inconnu	inconnu
6	Chad	inconnu	inconnu
7	Congo	inconnu	inconnu
8	Congo DR	inconnu	inconnu
9	Cameroon	inconnu	inconnu
10	Comoros	inconnu	inconnu
11	Central African Republic	inconnu	inconnu

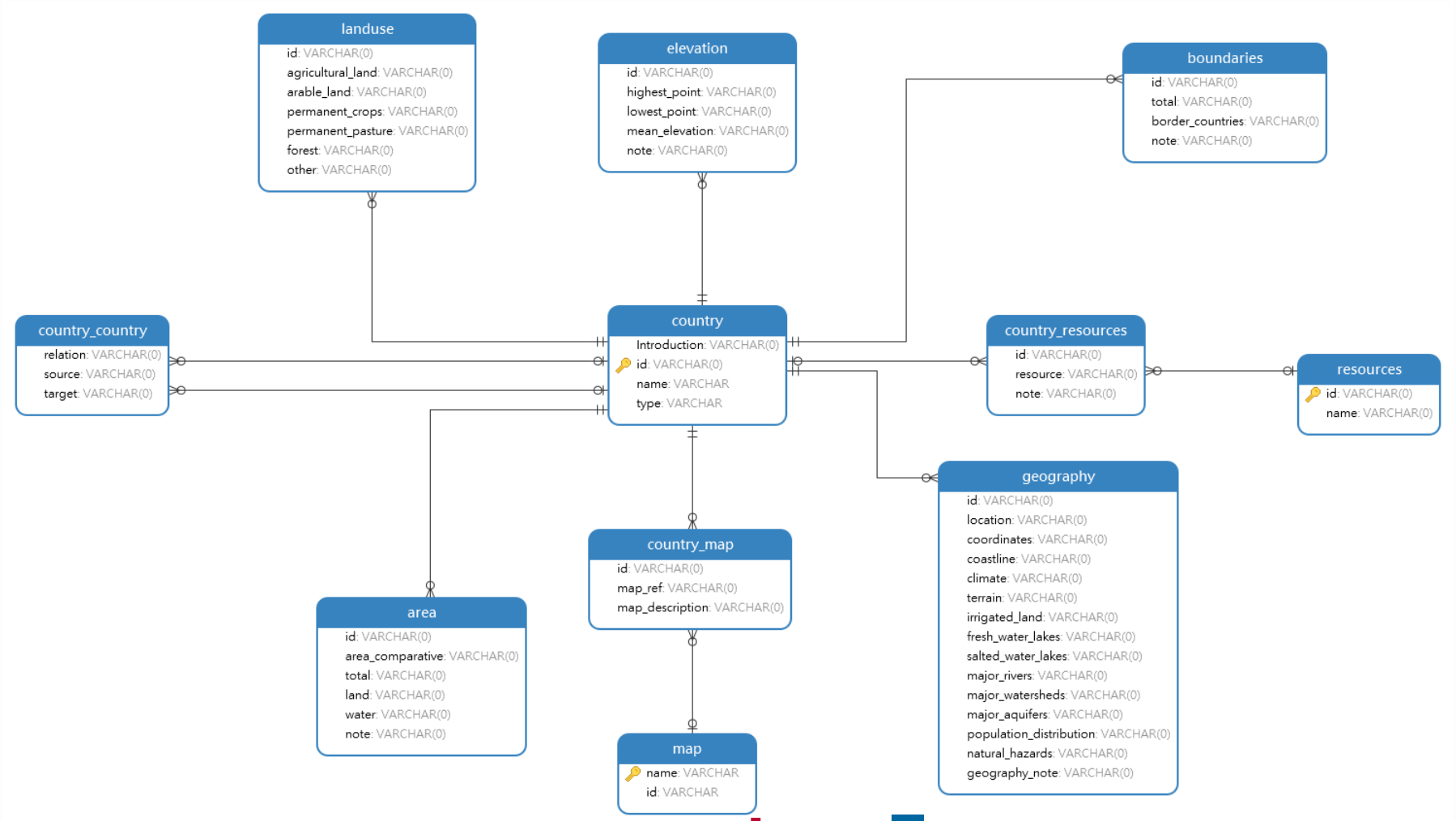
Les Object-Relational Mapping (ORM)



TD: brancher une base de données et l'interroger avec un ORM

TD Séance3

Relations entre classes



Relations one-to-many

```
class A(db.Model):
    cle_primaire = db.Column(db.String(100), primary_key=True)
    un_champ_de_la_table = db.Column(db.String(100))

    propriete_de_relation = db.relationship(
        "classe_liee",
        backref="classes_liees",
        lazy="dynamic"
     )
class classe_liee(db.Model):
    un_champ_de_la_table = db.Column(db.String(100))

    a_id = db.Column(db.String(100), db.ForeignKey('a.cle_primaire'))
```

Exercice

La relation entre les tables `country` et `area` est une relation one-to-many. A partir de la classe `Country()` écrite précédemment dans le cours, ajouter la relation avec la table `area` :

- écrire la classe `Area()`
- écrire la propriété de relation du côté de `Country()`
- ajouter la clé étrangère dans `Area()`

Correction

```
class Country(db.Model):
    __tablename__ = "country"
    id = db.Column(db.String(10), primary_key=True)
    ...
    areas = db.relationship(
        'Area',
        backref='areas',
        lazy=True)

class Area(db.Model):
    __tablename__ = "area"
    ...
    total = db.Column(db.String(100), primary_key=True)
    ...
    id = db.Column(db.String(100), db.ForeignKey('country.id'))
```

Interroger les relations one-to-many

```
# routes/generales.py /pays

for country in Country.query.all():
        ...
        print(country.areas)
return ...
```

Relations many-to-many

```
table_de_relation_a_vers_b = db.Table(
    "table_de_relation_a_vers_b",
    db.Column('a_id', db.String(100), db.ForeignKey('a.id'), primary_key=True),
    db.Column('b_id', db.String(100), db.ForeignKey('b.id'), primary_key=True))


class A(db.Model):
   id = db.Column(db.String(100), primary_key=True)
   un_champ_de_la_table = db.Column(db.String(100))

   bs = db.relationship('B',
        secondary=table_de_relation_a_vers_b,
        backref="bs")

class B(db.Model):
   id = db.Column(db.String(100), primary_key=True)
   un_champ_de_la_table = db.Column(db.String(100))
```

Exercice

Ecrire dans le modèle de données `factbook.py` la relation many-to-many entre `country` et `map`

Correction

```
# table de relation
country_map = db.Table(
    "country_map",
    db.Column('id', db.String(100), db.ForeignKey('country.id'), primary_key=True),
    db.Column('map_ref', db.String(100), db.ForeignKey('map.name'), primary_key=True)
)
```

Correction

```
class Country(db.Model):
    __tablename__ = "country"
    id = db.Column(db.String(10), primary_key=True)
    ...
    maps = db.relationship(
        'Map',
        secondary=country_map,
        backref="maps"
    )
   ...

class Map(db.Model):
    __tablename__ = "map"
    ...
    name = db.Column(db.Text, primary_key=True)
   ...
```

Interroger les relations many-to-many

```
# routes/generales.py /pays

for country in Country.query.all():
        ...
        print(country.maps)
return ...
```

Interroger les relations many-to-many

```
# routes/generales.py /pays

for country in Country.query.all():
        ...
        print(country.maps[0].name)
return ...
```

Exercices Séance3

- Au moins 1 (au choix) à l'issue de chaque séance (je ne prends en considération que le meilleur rendu par séance)
- Note globale à l'issue du cours (moyenne de la meilleure note de chaque séance)
- Exercice obligatoire noté à part (coeff 1):
  - Séance 3: *REV-3 : Ecrire un modèle de base de données (2)*

Cahier des charges

- Non noté dans l'UE5, mais essentiel pour une bonne réalisation de l'application finale
- Pour le 16 ou 22 janvier 2022 (choix des dates d'ici mi-janvier)
- Dépôt avec d'anciens devoirs : Chartes-TNAH
- Exemple cahier des charges pour décrire les sources de données et les fonctionnalités :exemple