

# Jour 11 - L'instruction is\_in et les pivots

L'instruction `is_in` permet de trouver les surfaces (de type `area`, cf. Jour 7) qui **contiennent** un point. C'est en quelque sorte l'*opération inverse* du filtre (`area`). Dans sa forme la plus simple, il suffit de passer les coordonnées d'un point :

```
is_in(48.8810863, 2.3830748);  
out;
```

Le résultat est l'ensemble des surfaces contenant ce point, visibles dans l'onglet Données. On y trouve les différents niveaux de limites administratives (en France: le pays, la région, le département, l'EPCI, la commune, l'arrondissement ou quartier), et des éléments d'occupation du sol.

Il n'est pas possible d'appliquer un filtre directement sur cette instruction, il est nécessaire de passer par un lot de données intermédiaire. L'exemple suivant sélectionne les limites administratives contenant le point :

```
is_in(48.8810863, 2.3830748) -> .a;  
area.a[boundary=administrative];  
out;
```

Un élément de type surface (`area`) *ne référence pas* les ways et nodes OSM, il n'est donc pas possible d'exporter sa géométrie directement. Il est nécessaire pour cela de **passer par l'objet OSM correspondant**, qui peut être soit un way fermé soit une relation (de type multipolygon ou boundary). Cela se fait au moyen des instructions `way(pivot)` et `rel(pivot)`.

Si l'on recherche une limite administrative, on peut se contenter de `rel(pivot)`. La requête suivante produit la commune qui contient le point :

```
is_in(48.8810863, 2.3830748) -> .a;  
rel(pivot.a)  
  [type=boundary]  
  [admin_level=8];  
out geom;
```

Si on recherche les éléments d'occupation du sol, il faut prendre en compte les ways et les relations :

```
is_in(48.8810863, 2.3830748) -> .a;  
(  
  way(pivot.a);  
  rel(pivot.a)[type!=boundary];  
);  
out geom;
```

L'instruction `is_in` peut également être utilisée sur un lot de données. **Seuls les nodes** de ce lot sont considérés. Voici une requête qui retourne les pierres runiques de France et les communes dans lesquelles elles se trouvent :

```
area[name=France] ->.a;  
node[historic=rune_stone](area.a);  
out;  
is_in;  
rel(pivot)  
  [type=boundary]  
  [admin_level=8];  
out geom;
```

Si le lot de données contient des ways ou des relations, il est nécessaire de passer par leurs nodes. Cela se fait avec l'instruction >; (cf. Jour 2), qui produit tous les nodes référencés par un lot de données.

La requête suivante trouve les communes de Vendée traversées par l'autoroute A87.

```
area  
  [admin_level=6]  
  [name="Vendée"] ->.dept;  
// Tracé de l'A87 en Vendée  
way  
  [highway]  
  [ref="A 87"]  
  (area.dept);  
out geom;  
  
// Communes traversées par ce tracé  
>;  
is_in;  
rel(pivot)  
  [boundary=administrative]  
  [admin_level=8]  
  (area.dept);  
out geom;  
  
{{style:  
relation {text:name}  
}}
```

© CC-by-sa Carto'Cité

From:

<http://wiki.cartocite.fr/> -



Permanent link:

[http://wiki.cartocite.fr/doku.php?id=tutoverpass:jour\\_11\\_l\\_instruction\\_is\\_in\\_et\\_le\\_filtre\\_pivot](http://wiki.cartocite.fr/doku.php?id=tutoverpass:jour_11_l_instruction_is_in_et_le_filtre_pivot)

Last update: **2021/01/25 17:14**

