Manipulation d'une base de données PostgreSQL avec les triggers / procédures stockées

A partir de la base tp et des relations 'villes' et 'routes':

1 Avec plpgsql

Créez une relation villesTrigger qui contient le schéma de villes avec un attribut supplémentaire, portionRoutiere, de type GEOMETY

```
create table villesTrigger
  ( id integer,
    nom text,
    portionRoutiere GEOMETRY
  );
```

Créez une procédure, constructionRoutiere, en plpgsql qui permet de stocker dans l'attribut portionRoutiere, la partie urbaine de la route qui traverse la ville considérée

```
CREATE FUNCTION constructionRoutiere () RETURNS TRIGGER AS
'
  DECLARE

    intersectionRoutiere geometry;

  BEGIN
    select into intersectionRoutiere ST_intersection(v.the_geom, r.the_geom)
    from   villes v, routes r
    Where  st_intersects (V.the_geom, r.the_geom)
          and v.id = NEW.id;

    NEW.portionRoutiere:=intersectionRoutiere;

    RETURN NEW;
  END;
'
LANGUAGE 'plpgsql';
```

Créez un trigger, trig_partieRoutiere qui sur une insertion dans villesTrigger vient remplir automatiquement l'attribut portionRoutiere à l'aide de la procédure constructionRoutiere.

```
CREATE TRIGGER trig_partieRoutiere BEFORE INSERT ON villesTrigger
  FOR EACH ROW
  EXECUTE PROCEDURE constructionRoutiere();
```

Chargez la relation villesTrigger à partir des tuples de la relation villes

```
insert into villesTrigger (id, nom)
  select id, nom
  from   villes;
```

Vérifiez que l'attribut portionRoutiere a bien été rempli.

```
select id, nom, ST_AsText (portionRoutiere) as portionRoutiere
 from   villesTrigger;
```

```
 id | nom |                    portionroutiere
----+-----+-----------------------------------------------------
  1  | V1  | LINESTRING(1 2,4 3.63636363636364)
  2  | V2  | LINESTRING(6 4.72727272727273,8 5.81818181818182)
  3  | V3  |
(3 rows)
```

Supprimez les éléments créés.

```
drop trigger trig_partieRoutiere on  villesTrigger;

drop function constructionRoutiere();

drop table villesTrigger;
```

2 Avec C

- Créez une relation routesTrigger qui contient le schéma de routes avec un attribut supplémentaire, longueur, de type réel.

```
create table routesTrigger
  ( id integer,
    nom text,
    longueur double precision
  );
```

- Créez un trigger, trig_longueurRoutiere, qui va lors de l'insertion d'un tuple dans routesTrigger va déclencher l'exécution de la fonction constructionlongueur.

```
CREATE TRIGGER trig_longueurRoutiere AFTER INSERT ON routesTrigger
  FOR EACH ROW
  EXECUTE PROCEDURE constructionLongueur();
```

- Créez une fonction, constructionlongueur, en C qui va venir remplir l'attribut longueur avec la longueur de la route associée au tuple.

- Chargez la relation routesTrigger à partir des tuples de la relation routes.

```
insert into routesTrigger (id, nom)
  select id, nom
  from   routes;
```

- Vérifiez que l'attribut longueur a bien été rempli.

```
select id, nom, longueur
from   routesTrigger;

 id | nom | longueur
----+-----+-----------
  1 |   R1  | 15.529964
  2 |   R2  |         2
  3 |   R3  |         2
(3 rows)
```

Supprimez les éléments créés.

```
drop trigger trig_longueurRoutiere on  routesTrigger;
drop function constructionLongueur();
drop table routesTrigger;
```