Soit le schema :

Country(countrycode, name_country , continent, region, population_country, lifeexpectancy, capital, governmentform, gnp)
Language(countrycode, language, isofficial, percentage)
City(id, name_city, countrycode, district, population_city)

avec contraintes de clés étrangeres :
city.countrycode subset country.countrycode
langauge.countrycode subset countrycountrycode
country.capital subset city.id

1- Ecrire une requete SQL qui renvoie, pour chaque continent, la région avec le plus grand nombre de langues officielles parlées dans la région ?

sol 1 : on peut faire une table intermédiaire qui donne, pour chaque région (de tous continents), le nombre de langues officielles parlées dans cette région. Afin d'utiliser ensuite cette table intermédiaire, on précise le continent à coté de la région.
(si la table est cohérente, un unique continent est associé à une région donnée, mais il n'est pas utile de faire cette hypothese, grâce au group by … En effet, si une région apparait dans deux continents différents, le group by va scinder la région en "sous-régions" selon le continent d'appartenance, et le nombre de langues parlées peut alors varier selon quel morceau de la région est considéré)

Select region, continent, COUNT(distinct language) from country natural join language where isofficial group by region, continent ;

… et donc :

```
WITH Langues AS
( select region, continent, COUNT(distinct language) as nbe from country natural join language where isofficial group by region, continent )
select L1.continent, L1.region from Langues L1
where L1.nbe=(select MAX(L.nbe) from Langues L where L.continent=L1.continent ) ;
```

sol 2 : sans table temporaire

```
select continent, region from country c natural join language
where isofficial
group by continent, region
having COUNT(distinct language)>= ALL (select COUNT(distinct l2.language) from country natural join language l2
group by l2.continent, l2.region having l2.continent=c.continent) ;
```

remarque : on peut aussi utiliser =(select MAX(foo.nbe) FROM (select COUNT(…) AS nbe from coutry …) AS foo) ;
plutot que >= ALL (select COUNT(…) from country …) ; comme ci-dessus
Mais pour rappel, on ne peut pas faire (select MAX (COUNT (…)) from …).

Sol 3 : autre solution sans table temporaire

```
Select distinct c.continent, c.region from from country natural join language as c
Where not exists (select * from country natural join language as c2 where c2.continent=c.continent and c2.isofficial
group by c2.region having COUNT(distinct c2.language) >
(Select count(distinct c3.language) from country natural join language as c3 where
```

c3.continent=c.continent and c3.isofficial
group by c3.region having c3.region=c.region)) ;

Erreur courante

Beaucoup vu la requête suivante :

Select continent, region from country natural join language
where isofficial
Group by region
Having COUNT(distinct language)=(select MAX(COUNT(distinct c2.language)) from country natural join language as c2
Where c2.isofficial group by c2.region having c2.continent=continent) ;

… pour rappel on ne peut pas emboiter deux opérateurs d'agrégation : MAX(COUNT(…)) renverra un msg d'erreur.
Par ailleurs soyez conscient que la requete ci-dessus sous-entend qu'une région donnée n'apparait que dans un continent.

(quand bien meme : peut-on demander un select continent, si on regroupe par region seulement ?)

Q2. Pays avec moins de 10 villes et tels que la capitale a une population > 30% de la population du pays ?
+ pays avec 10 villes ou plus, et la capitale a >30% de la population des dix villes les plus peuplées, réunies ?

La requête était d'un niveau trop compliqué pour le contrôle. Voici une solution correcte, mais très longue (il faut en effet veiller à bien remplacer les …, ainsi que la clause Ci.id <> Cj.id pour tout 0<i<j<11,par du code SQL) :

```
WITH MoinsDix AS (SELECT countrycode
                  FROM City
                  GROUP BY countrycode
                  HAVING COUNT(*) < 10)
     PlusDix AS (SELECT countrycode
                  FROM City
                  GROUP BY countrycode
                  HAVING COUNT(*) > 10)
    DixCity AS (SELECT countrycode, C1.population_city+…+C10.population_city as nb
                  FROM City C1, City C2, City C3, City C4, City C5, City C6, City C7, City C8, City C9, City C10
                  WHERE C1.countrycode=C2.countrycode
                  AND …
                  AND C9.countrycode=C10.countrycode
                  AND Ci.id <> Cj.id pour tout 0<i<j<11
                  AND NOT EXISTS (SELECT * FROM City C11 WHERE C11.countrycode=C1.countrycode AND (C11.population_city > C1.population_city OR … OR C11.population_city > C10.population_city) AND (C11.id<>C1.id AND … C11.id <> C10.id))

SELECT name_country
FROM Country NATURAL JOIN MoinsDix, City
WHERE City.id=Country.capital
```

```
AND population_city > 0,3*population_country
                    UNION
SELECT name_country
FROM Country NATURAL JOIN PlusDix NATURAL JOIN DixCity, City
WHERE City.id=Country.capital
AND population_city > 0,3*nb ;
```

Il est possible d'écrire les choses de manière plus compacte en faisant appel à des fonctions prédéfinies de Postgres, mais nous ne détaillerons pas ce genre de solution ici. Si vous êtes curieux vous pouvez consulter https://www.postgresql.org/docs/current/tutorial-window.html (ou dans le contexte d'autres SGBDs https://stackoverflow.com/questions/54612392/sql-sum-of-highest-n-values-in-each-group, ou encore http://www.silota.com/docs/recipes/sql-top-n-aggregate-rest-other.html).

Les solutions à base de ORDER BY et LIMIT qui vont être présentées maintenant ont été acceptées comme correctes,mais elles ne renvoient pas le résultat désiré. Pour comprendre pourquoi, considérez le code suivant :

```
user=# create table t (a int, b int);
CREATE TABLE
user=# insert into t values (1,2);
INSERT 0 1
user=# insert into t values (1,3);
INSERT 0 1
user=# select * from t;
 a | b
---+---
 1 | 2
 1 | 3
(2 rows)
user=# select a, sum(b), count(*) from t group by a order by a desc limit 1;
 a | sum | count
---+-----+-------
 1 |   5 |     2
(1 row)
```

On constate que les agrégats ont opéré sur l'ensemble des tuples et non uniquement sur le premier tuple (sinon on aurait obtenu (1,3,1)). Le ORDER BY et le LIMIT sont toujours interprétés à la fin de l'évaluation et donc après le SELECT…

Solutions acceptées comme correctes bien que buggées :

—> on peut éventuellement faire un WITH qui donne les pays, avec la population des dix villes les plus peuplées, le nombre de villes du pays (dans l base), la population du pays, et la population de la capitale. On peut obtenir cette table temporaire en une fois, à condition de faire une jointure, ou sinon en plusieurs fois.

```
WITH  AUX(countrycode, pop_villes, nbe_villes)  AS
(select countrycode, SUM(population_city), COUNT(*) from city group by
countrycode
ORDER BY population_city DESC LIMIT 10),
Pays(countrycode, nom_pays, pop_villes, nbe_villes, pop_pays, pop_cap) AS
(select c.countrycode, c.name_country, c.pop_villes, c.nbe_villes,
```

```
c.population_country as pop_pays, population_city as pop_cap
from (AUX natural join country) as c Join city on c.capital=city.id )

select nom_pays from Pays where pop_cap>0.3 * pop_pays
union
select nom_pays from Pays where nbe_villes>9 and pop_cap>0.3 * pop_villes ;
```

Autre solution :

```
select name_country as nom from country inner join city on country.capital=city.id
where 10*city.population_city> 3* country.population_country
union
select nom from (select c2.name_country as nom, c2.countrycode,
SUM(c2.population_city) as pop_villes
 from country natural join city c2
group by c2.countrycode having COUNT(distinct c2.id)>=10
and 3*pop_villes < 10* (select c3.population_city from city c3 where c3.id=c2.capital)
ORDER BY c2.population_city DESC LIMIT 10) as foo ;
```

3/ Donner pour chaque région, le pays avec le plus haut gnp par habitant ?

```
select distinct region, name_country from country c1 where not exists
(select * from country c2 where c2.region=c1.region and
c2.gnp/c2.population_country > c1.gnp / c1.population_country) ;
```

ou bien :

```
WITH AUX AS
(select region, countrycode, name_country as nom, gnp/population_country as
gnp_hab from country
where population_country>0)
select region, countrycode from AUX
where aux.gnp_hab=(select MAX(a2.gnp_hab) from AUX a2
where a2.region=aux.region) ;
```

3b/ Donner les régions qui maximisent l'écart (dans la région) entre lifeexpectancy la plus haute et la plus basse ?

```
WITH table(region, ecart) AS
(select region, MAX(lifeexpectancy)- MIN(lifeexpectancy)
from country group by region)
select distinct region from table
where ecart=(select MAX(ecart) from table) ;
```

ou alors :

```
select region from country
group by region
having MAX(lifeexpectancy)-MIN(lifeexpectancy)>= ALL(select
MAX(lifeexpectancy)-MIN(lifeexpectancy) from country group by region) ;
```

Q4 - Régions où il n'existe qu'une seule forme de gouvernement ?

Requête avec op. d'agrégat :

```
select region from country group by region
having COUNT(distinct governmentform)=1 ;
```

Requete sans COUNT et sans ALL :

```
select distinct c1.region from country c1
where not exists (select * from country c2 where c2.region=c1.region
and c2.governmentform <> c1.governmentform ) ;
```

Remarque : les deux solutions ci-dessus renvoient aussi les régions où il existe une seule forme de gouvernement, et aussi des pays dans la région avec un NULL comme governmentform. Comment modifieriez-vous les requetes, si on ne veut pas de ces régions (pour lesquelles on n'est pas certains qu'il y a unicité de la governmentform dans la région) ?

Question supplémentaire

Schema :
Films(titre, realisateur, duree)
Seances(idseance, cinema, titre, jour, heure_debut)

Question :
quels sont les cinemas où plus de la moitié des films diffusés, sont des films de Varda, et tels qu'au moins un tiers des films réalisés par Varda, soient diffusés dans le cinema ?

On veut comparer trois quantités: N1 le nombre de films de Varda dans le cinema c
N2 le nombre de films (distincts) diffusés dans le cinema c,
et N3 le nombre de films réalisés par Agnes Varda.
(la contrainte de clé étrangère (qui etait donnée par l'énoncé) fait qu'on aura toujours N1 N3)

On veut les cinémas c tels que 3*N1 > N3 (seconde condition) et tels que 2*N1 >N2 (première condition).

Attention : ceux qui dans leur requête ont fait appel à la quantité N'2 = films de réalisateurs autres que Varda diffusés dans le cinema c,
la condition devient N1>N'2 … et non N1 > 0.5 N'2 …

```
select cinema from Seances S natural join Films
where S.realisateur='Varda'
group by cinema
HAVING 2*COUNT(distinct S.titre)>(select COUNT(distinct S2.titre) from Seances
S2 group by S2.cinema having S2.cinema=S.cinema)
AND 3*COUNT(distinct S.titre)>(select COUNT(distinct F.titre) from Films F where
F.realisateur='Varda') ;
```

Autre solution possible [la troisième table est franchement peu utile,
préférer faire un (select COUNT(distinct titre) from Films where realisateur='Varda'),
à l'endroit où la table "Real" est utilisée…] :

```
WITH AUXV AS
(select cinema as cine, COUNT(distinct titre) as nbe_Varda from Seances natural
join Films
where realisateur='Varda' group by cinema),
Aux AS
(select cinema as cine, COUNT(distinct titre) as nbe_tot from Seances
group by cinema),
Real AS
(select realisateur, COUNT(distinct titre) as total from Films group by realisateur)
```

```
select cine from AUXV join Aux using cine
where nbe_Varda>0.5 * nbe_tot
and 3*nbe_Varda > (select total from Real where realisateur='Varda') ;
```