ITC2

Corrigé TP SQL

C. MULLAERT

Lycée Saint-Louis

Année 2023-2024

Requêtes utilisant les fonctions d'agrégation

Donner le nombre de pays où l'espérance de vie est supérieure à 70 ans.

```
SELECT COUNT(*) FROM country WHERE Lifeexpectancy>=70
```

Donner, pour chaque code pays à trois lettre, le nombre de langues parlées dans ce pays.

```
SELECT countryCode, COUNT(*) FROM countryLanguage
GROUP BY countryCode
```

Afficher la liste des régions d'Europe avec, pour chaque région, le nombre de pays qui la compose.

```
SELECT Region, COUNT(Name) FROM country
WHERE Continent='Europe' GROUP BY Region
```

Donner, pour chaque code pays à trois lettre, la moyenne du nombre d'habitants des villes de ce pays présentes dans la base.

```
SELECT countryCode, AVG(Population) FROM city
GROUP BY countryCode
```

Donner, pour chaque code pays à trois lettre, le pourcentage de la population parlant une langue officielle

```
SELECT countryCode, SUM(Percentage) FROM countryLanguage
WHERE IsOfficial='T' GROUP BY countryCode
```

Afficher pour chaque continent, le PIB le plus important

```
SELECT continent, MAX(GNP/Population ) from Country
GROUP BY continent
```

Jointures

Afficher la liste des noms des pays avec leur capitale

```
SELECT Co.Name, Ci.Name
FROM country AS Co JOIN city AS Ci ON Co.Capital = Ci.ID
```

ou

```
SELECT Co.Name, Ci.Name from city AS Ci JOIN country AS Co
ON Code=Countrycode WHERE Id=Capital
```

Afficher pour chaque couple (Continent, Région) le nombre de langues différentes parlées dans les pays de cette région.

```
SELECT Continent, Region, COUNT(*) FROM country AS Co
JOIN countryLanguage AS Cl ON Co.Code = Cl.countryCode
GROUP BY Continent, Region
```

Afficher la liste des noms de pays dont au moins 50% de la population parle une langue officielle.

```sql
SELECT Co.Name
FROM country AS Co JOIN countryLanguage AS Cl
ON Co.Code=Cl.countryCode
WHERE IsOfficial='T'
GROUP BY Co.Name
HAVING SUM(Cl.Percentage)>50
```

ou

```sql
SELECT Name
FROM country JOIN countrylanguage ON Code=countryCode
WHERE IsOfficial='T'
GROUP BY Name
HAVING SUM(Percentage)>50
```

Sous-requêtes

Afficher la liste des villes de France

```
SELECT Name FROM city WHERE countryCode
IN (SELECT Code FROM country WHERE Name='France')
```

ou

```
SELECT Ci.Name FROM city AS Ci JOIN country AS Co
ON Ci.countryCode=Co.Code WHERE Co.Name='France'
```

Afficher la liste des noms des pays Européens dont la population est inférieure à la moyenne mondiale.

```sql
SELECT Name FROM country WHERE Continent='Europe'
AND Population < (SELECT AVG(Population) FROM country)
```

Afficher, pour chaque code pays, le nombre de villes dont la population dépasse la population moyenne des villes de France

```sql
SELECT countryCode, COUNT(*) FROM city
WHERE Population >
        (SELECT AVG(Ci.Population) FROM city AS Ci
         JOIN country AS Co ON Ci.countryCode=Co.Code
         WHERE Co.Name='France' )
GROUP BY countryCode
```

Afficher la liste des noms des pays Européens ainsi que l'écart entre leur PIB par habitant et la moyenne Européenne.

```
SELECT Name, GNP/Population*1000 -
        (SELECT AVG(GNP/Population*1000) FROM country
         WHERE Continent='Europe')
FROM country WHERE Continent='Europe'
```

Afficher pour chaque continent, le pourcentage de pays pour lesquels le PIB par habitant est supérieur à la moyenne du continent

```
SELECT Co1.Continent, COUNT(*)*100/
        (SELECT COUNT(*) from country AS Co2
         where Co2.Continent=Co1.Continent)
FROM country AS Co1
WHERE Co1.GNP/Co1.Population >
        (SELECT AVG(Co3.GNP/Co3.Population)
         FROM country AS Co3
         WHERE Co3.Continent=Co1.Continent )
GROUP BY Co1.continent;
```

Afficher la liste des villes dont le nom correspond à au moins deux pays. On pourra, dans un second temps afficher simultanément les codes de deux pays correspondant.

```
SELECT DISTINCT p.Name FROM city AS p WHERE p.Name IN
        (SELECT q.Name FROM city AS q WHERE q.ID<>p.ID)
```

ou mieux :

```
SELECT Name FROM city GROUP BY Name HAVING COUNT(*)>1
```

ou alors (autojointure) :

```
SELECT DISTINCT p.Name FROM city AS p JOIN city AS q
ON p.Name=q.Name AND p.ID < q.ID
```

ou bien

```
SELECT ci1.Name FROM city AS ci1 JOIN city AS ci2
WHERE ci1.Name=ci2.Name AND ci1.ID!=ci2.Id
GROUP BY ci1.Name
```

avec les codes :

```
SELECT p.Name, p.countryCode, q.countryCode
FROM city AS p JOIN city AS q
ON p.Name=q.Name AND p.ID < q.ID
```

Afficher, pour chaque région, le pays le plus peuplé.

```
SELECT Co2.Region,
        (SELECT Name FROM country AS Co
         WHERE Co.Region=Co2.Region
         ORDER BY Population DESC LIMIT 1)
FROM country AS Co2 GROUP BY Region
```

ou

```
SELECT C1.Region,
        (SELECT Name FROM country AS C2
         WHERE C2.Region=C1.Region
         AND Population =
           (SELECT MAX(Population) FROM country AS C3
                WHERE C3.Region=C1.Region ) LIMIT 1)
FROM country AS C1 GROUP BY Region
```

Afficher, pour chaque région, le deuxième pays le plus peuplé.

```
SELECT Co2.Region,
        (SELECT Name FROM country AS Co
         WHERE Co.Region=Co2.Region
         ORDER BY Population DESC LIMIT 1 OFFSET 2)
FROM country AS Co2
GROUP BY Region
```

Écrire une requête SQL qui donne le nombre de simulations effectuées pour chaque nombre de dimensions de l'espace de simulation.

```
SELECT SI_DIM,COUNT(*) FRON SIMULATION GROUP BY SI_DIM
```

Écrire une requête SQL qui donne, pour chaque simulation, le nombre de rebonds enregistrés et la vitesse moyenne des particules qui frappent une paroi.

```
SELECT SI_NUM,COUNT(*),AVG(RE_VIT)
FROM REBOND GROUP BY S.SI_NUM
```

Ecrire une requête SQL qui, pour une simulation n donnée, calcule, pour chaque paroi, la variation de quantité de mouvement due aux chocs des particules sur cette paroi tout au long de la simulation.

On se rappellera que lors du rebond d'une particule sur une paroi la composante de sa vitesse normale à la paroi est inversée, ce qui correspond à une variation de quantité de mouvement de $2m|v_\perp|$ où m désigne la masse de la particule et $v_\perp$ la composante de sa vitesse normale à la paroi.

```
SELECT RE_DIR,SUM(2*PA_M*RE_VP) FROM REBOND R
JOIN PARTICULE P ON R.PA_NUM=P.PA_NUM
WHERE SI_NUM=n GROUP BY RE_DIR
```

"Quels sont le numéro d'identification et le nom de site des bouées localisées en Méditerrannée ?"

```
SELECT idBouee, nomSite FROM Bouee
WHERE localisation = "Mediterranee"
```

"Quel est le numéro d'identification des bouées où il n'y a pas eu de tempêtes ?"

```sql
SELECT idBouee FROM Bouee
EXCEPT
SELECT idBouee FROM Tempete
```

ou (mais les points ne sont pas assurés)

```sql
SELECT idBouee AS id FROM Bouee WHERE
       (SELECT COUNT(*) FROM Tempete WHERE idBouee=id) = 0
```

ou (mais les points ne sont pas assurés)

```sql
SELECT idBouee AS id FROM Bouee
       WHERE idBouee NOT IN (SELECT idBouee FROM Tempete)
```

"Pour chaque site, quelle est la hauteur maximale enregistrée lors d'une tempête ?"

```
SELECT nomSite, MAX(Hmax) FROM Bouee JOIN Tempete
ON Bouee.idBouee=Tempete.idBouee GROUP BY nomSite
```

ou avec des alias

```
SELECT nomSite, MAX(Hmax) FROM Bouee AS B
JOIN Tempete AS T ON B.idBouee=T.idBouee
GROUP BY nomSite
```

Écrire une requête SQL qui renvoie les identifiants des ensembles auxquels appartient le point de coordonnées (a, b).

```
SELECT idensemble FROM MEMBRE JOIN POINTS
ON idpoint=id WHERE x=a AND y=b
```

Ecrire une requête SQL qui renvoie les coordonnées des points qui appartiennent à l'intersection des ensembles d'identifiants i et j.

```
(SELECT x,y FROM   POINTS JOIN MEMBRE ON idpoint=id
WHERE idensemble=i)
INTERSECT
(SELECT x,y FROM   POINTS JOIN MEMBRE ON idpoint=id
WHERE idensemble=j)
```

ou (mais les points ne sont pas assurés)

```
SELECT x,y FROM
        (SELECT id,x,y FROM   POINTS JOIN MEMBRE
         ON id=idpoint WHERE idensemble=i) AS ENS1
JOIN MEMBRE ON id=idpoint WHERE  idensemble=j
```

ou avec une jointure de 3 tables (mais les points ne sont pas assurés)

```
SELECT x, y FROM POINTS
JOIN MEMBRE AS M1 ON id = M1.idpoint
JOIN MEMBRE AS M2 ON id = M2.idpoint
WHERE M1.idensemble = i AND M2.idensemble = j
```

Écrire une requête SQL qui renvoie les identifiants des points appartenant à au moins un des ensembles auxquels appartient le point de coordonnées (a, b).

```sql
SELECT M1.idpoint FROM MEMBRE as M1
JOIN MEMBRE as M2 ON M1.idensemble = M2.idensemble
JOIN POINTS ON POINTS.id = M2.idpoint
WHERE x = a AND y = b
```

ou

```sql
SELECT idpoint FROM MEMBRE WHERE idensemble IN
        (SELECT idensemble FROM MEMBRE JOIN POINTS
         ON idpoint=id WHERE x=a AND y=b)
```

ou

```
SELECT M1.idpoint FROM MEMBRE AS M1 JOIN
        (SELECT M2.idensemble AS id2 FROM MEMBRE AS M2
         JOIN POINTS ON M2.idpoint=id WHERE x=a AND y=b)
         AS T
ON M1.idensemble=T.id2
```