

Dans la suite, on dispose des tables décrites [figure 13.1](#). On marque par  $\star$  les colonnes qui forment la clef primaire de la table.

Name	Code $\star$	Capital	Province	Area	Population
				Surface du pays en km <sup>2</sup>	

(a) Table `country`

Country $\star$	Continent $\star$	Percentage
Clef étrangère pour <code>country.Code</code>		Pourcentage de la surface du pays dans le continent

(b) Table `encompasses`

Name $\star$	Country $\star$	Province	Population	Longitude	Latitude
	Clef étrangère pour <code>country.Code</code>				

(c) Table `city`

Country $\star$	Language $\star$	Percentage
Clef étrangère pour <code>country.Code</code>		Pourcentage de la population du pays qui parle la langue

(d) Table `spoken`

Country $\star$	GDP	Agriculture
Clef étrangère pour <code>country.Code</code>	Le PIB en million de dollars	La part de l'agriculture dans le PIB, en %
	Industry	Inflation
	La part de l'industrie dans le PIB	Le taux d'inflation
	Unemployment	
	Le taux de chômage	

(e) Table `economy`

FIGURE 13.1 – Structure de la base de données MONDIAL

## 13.1 Agrégations

SQL permet de regrouper les résultats d'une requête grâce au mot clef `GROUP BY`. Par exemple, on peut regrouper par continent dans la table `ENCOMPASSES` en écrivant `GROUP BY Continent`. Ensuite, les fonctions d'agrégation vues au TP précédent permettent de faire des calculs *au sein chaque groupe*. Voici par exemple comment obtenir le nombre de pays de chaque continent

```
SELECT continent, COUNT(*)
FROM encompasses
GROUP BY continent;
```

Ou encore, avec un exemple plus compliqué, voici comment obtenir la surface totale de chaque continent

```
SELECT e.continent, SUM(c.area * e.percentage)
FROM encompasses as e JOIN country as c ON c.code = e.country
GROUP BY e.continent;
```

Attention, une fois qu'on regroupe selon une certaine colonne, on ne peut adresser les autres colonnes que par l'intermédiaire des fonctions d'agrégation. Ainsi la requête suivante *ne veut rien dire et n'est pas correcte*

```
SELECT e.continent, c.name -- NON on ne peut pas sélectionner directement c.name
FROM encompasses as e JOIN country as c ON c.code = e.country
GROUP BY e.continent;
```

Ensuite, il est possible de filtrer les groupes avec le mot clef `HAVING`. Par exemple, on peut ne sélectionner que les continents d'une surface plus grande que 10 000 000 km<sup>2</sup> avec

```
SELECT e.continent
FROM encompasses as e JOIN country as c ON c.code = e.country
GROUP BY e.continent
HAVING sum(c.area * e.percentage) > 10000000;
```

Notez la différence entre **WHERE** et **HAVING** : le premier filtre *les lignes* avant de former les groupes, et le second filtre *les groupes* après les avoir formés. La forme générale d'une requête est donc

```
SELECT column
FROM table
WHERE ligne_condition
GROUP BY column
HAVING groupe_condition
```

## 13.2 Sous-requêtes

Le résultat d'une requête est lui-même une table qui peut être elle-même imbriquée dans une nouvelle requête. Par exemple, on peut récupérer la surface moyenne des pays avec la requête

```
SELECT avg(area) FROM country;
```

En réutilisant la requête précédente on peut ainsi obtenir les pays dont la surface est plus grande que la moyenne :

```
SELECT name
FROM country
WHERE area > (SELECT avg(area) FROM country);
```

Les requêtes imbriquées peuvent aussi être utilisées dans la clause **FROM table** à la place de **table**, mais dans cas il faut parfois donner un nom à la table et ses colonnes avec **AS**. Par exemple, la requête suivante compte le nombre de langues parlées dans chaque pays

```
SELECT count(*) FROM spoken GROUP BY country;
```

On réutilise cette requête pour obtenir le plus grand nombre de langues parlées dans un seul pays. Notez qu'on donne un nom à la colonne **count(\*)** pour pouvoir s'y référer dans le reste de la requête.

```
SELECT MAX(n_language)
FROM (SELECT count(*) as n_language FROM spoken GROUP BY country) as count_table;
```

Le mot clef **IN** peut être utilisé pour tester l'appartenance à une table (la table étant, en général, le résultat d'une requête imbriquée). Par exemple, on peut s'inspirer des requêtes ci-dessus pour obtenir le nombre de langues parlées dans les pays *dont la surface est inférieure à la moyenne mondiale*

```
SELECT country, count(*)
FROM spoken
WHERE country IN (
    SELECT code
    FROM country
    WHERE area < (SELECT avg(area) FROM country)
)
GROUP BY country;
```

Vous avez maintenant toutes les clefs pour finir les exercices suivants (les deux premiers sont des rappels du TP précédent). Si vous y arrivez, félicitations vous maîtrisez l'ensemble du programme de SQL de BCPST.

### Exercice 1 *Requêtes de base*

Rédiger une requête SQL pour obtenir

- La liste des pays dont la population excède 60M habitants
- La même liste triée par ordre alphabétique
- La liste des pays et de leurs populations respectives, triée par ordre croissant de population
- Le nom des dix pays ayant la plus petite superficie
- Le nom des dix suivants

### Exercice 2 *Jointures*

Rédiger une requête SQL pour obtenir

- Le nom des pays qui sont à cheval sur plusieurs continents
- Les pays du continent américain qui comptent moins de 10 habitants par km<sup>2</sup>
- Les capitales européennes situées à une latitude supérieure à 60°

**Exercice 3**    *Fonctions d'agrégation*

- a) Donner la liste ordonnée des dix langues parlées dans le plus de pays différents.
- b) Quelles sont les langues parlées dans exactement 6 pays ? Et de quel pays s'agit-il ?
- c) Quelles sont les langues parlées par moins de 30 000 personnes dans le monde ?
- d) Quelles sont les 5 langues les plus parlées en Afrique ? Préciser pour chacune d'elle le nombre de personnes qui la parlent.

**Exercice 4**    *Sous-requêtes*

- a) Déterminer les pays majoritairement agricoles dont le taux de chômage est inférieur à la moyenne mondiale.
- b) Déterminer pour chaque continent le pays au taux d'inflation le plus faible parmi les pays majoritairement industriels.
- c) Déterminer les pays dans lesquels on parle le plus de langues.
- d) Quel est le continent le plus riche, en terme de PIB ?