SQL – EXERCICES (1)

1. VILLES DE FRANCE

La base de données contient la table ***ville*s** dont le modèle est donné au dessous. La table ***departements*** permet de faire le lien entre le numéro (id_departement) et le nom (nom_departement) d´un département français.

Table *villes*

Field	Type	Null	Key
id	mediumint(8) unsigned	NO	PRI
id_departement	varchar(3)	YES	MUL
nom	varchar(45)	YES	MUL
code_postal	varchar(30)	YES	MUL
population	mediumint(10) unsigned	YES	
densite	int(11)	YES	
surface	float	YES	
longitude	float	YES	MUL
latitude	float	YES	
ville_zmin	mediumint(4)	YES	
ville_zmax	mediumint(4)	YES	

Table *departements*

Field	Type	Null	Key
code_departement	varchar(3)	NO	PRI
nom_departement	varchar(23)	NO	
code_region	int(11)	NO	
nom_region	varchar(26)	NO	

Ecrivons les requêtes SQL qui permettent d´afficher:

1) Les **100 premiers** enregistrements de la table.

.....

2) Les villes dont le nom se termine par un 'W'.

.....

3) Le nom des villes de plus de **100 000 habitants**.

.....

4) Le nom des **10 villes les plus peuplées** affichées par ordre décroissant de la population.

.....

5) Toutes les informations sur les villes du **département de la Gironde**.

.....

6) Le **nombre de villes** enregistrées dans la base.

.....

7) Le **nombre de villes** du département de la **Dordogne** enregistrées dans la base.

.....

8) La **population moyenne** des villes de France.

.....

9) Afficher le nom, et la latitude de la **ville de Bonifacio** (en Corse).

.....

10) Afficher le nom, le département (*id_departement*) et la latitude des **villes plus au Sud de Bonifacio**.

> ☞ *Cette méthode consiste en une sous requête.*

.....

.....

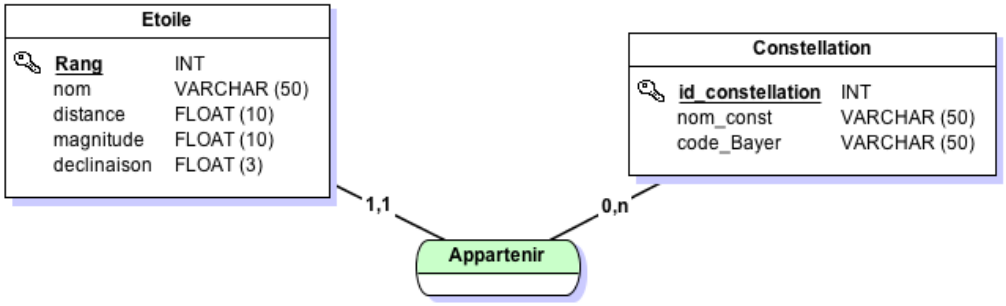
Plus difficile...

11) Les **populations totales** de chaque département et leur id correspondant (*id_departement*) classées en ordre décroissant de population.

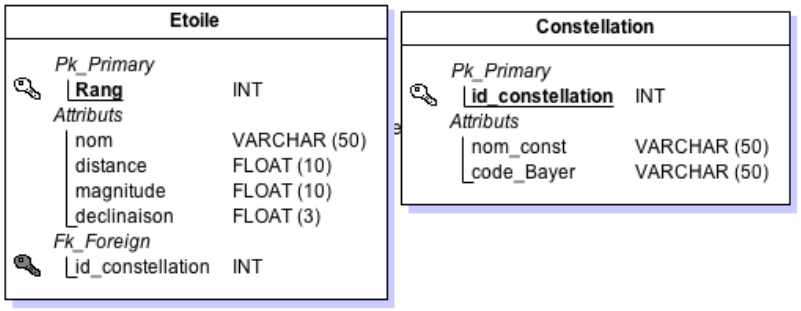
> ☞ La fonction d´agrégation **SUM(*colonne1*)** calcule la somme des données de la *colonne1*. On peut regrouper cette somme par valeurs enregistrées dans une autre colonne (*colonne2*) avec la clause **GROUP BY *colonne2***. Cette clause n´est pas au programme.

2. LES ETOILES

La base de données contient les tables ***Etoile*** qui contient les 35 étoiles les plus brillantes du ciel ***et Constellation*** qui proviennent du MCD suivant:



On obtient le *modèle relationnel*:



1) Ecrire la **représentation textuelle** des tables du modèle relationnel:

☞ Pour qu´une étoile soit **visible** dans le ciel, il faut que sa déclinaison (δ) respecte le critère suivant:

$$\delta > \varphi - 90$$

☞ Les étoiles qui respectent le critère suivant ne se couchent jamais, on dit qu´elle sont **circumpolaires**:

$$\delta > 90 - \varphi$$

☞ A Lisbonne: $\varphi = 38.7°$

Ecrivons les requêtes SQL qui permettent d´afficher:

2) Le contenu de chacune des deux tables:

.....

.....

3) Le nom et l´id de la constellation d´Orion:

.....

4) Le nom des étoiles de la constellation d´Orion enregistrées dans la base:

.....

5) Le nom et la déclinaison des étoiles **visibles** dans le ciel de Lisbonne:

.....

6) Le nom, la déclinaison et l´id de la constellation des **étoiles circumpolaires** du ciel de Lisbonne:

.....

*A l´aide de **jointures** écrivons les requêtes SQL qui permettent d´afficher:*

7) La jointure complète des deux tables.

.....
.....
.....

8) Le nom des étoiles de la base avec le nom de la constellation à laquelle elles appartiennent.

.....
.....
.....

9) Reprendre la question 6) en affichant en plus le **nom de la constellation** à laquelle appartiennent ces deux étoiles.

.....

.....

.....

.....

10) Le nombre d´étoiles par constellation de la base affichées par ordre décroissant du nombre.

.....

.....

.....

.....

.....