**SQL – EXERCICES (1) CORRECTION**

08

**Automne**

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***



**Table *departements***



*Ecrivons les requêtes SQL qui permettent d´afficher:*

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

**SELECT \* FROM villes LIMIT 100**

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

**SELECT nom FROM villes WHERE nom LIKE('%W')**

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

**SELECT nom FROM villes WHERE population>100000**

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

**SELECT nom FROM villes ORDER BY population DESC LIMIT 10**

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

***On peut faire une recherche préalable dans la table*** *departements* ***pour connaître le numéro de la Gironde.*..
SELECT \* FROM villes WHERE id\_departement=33**

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

**SELECT COUNT(\*) FROM villes**

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

***On peut faire une recherche préalable dans la table*** *departements* ***pour connaître le numéro de la Dordogne.*..
SELECT COUNT(\*) FROM villes WHERE id\_departement=24**

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

**SELECT AVG(population) FROM villes**

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

**SELECT nom, latitude FROM villes WHERE nom='Bonifacio'**

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.*

**SELECT nom, id\_departement,latitude FROM villes WHERE latitude < (SELECT latitude FROM villes WHERE nom='Bonifacio')**

*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.

**SELECT SUM(population), id\_departement FROM villes
GROUP BY id\_departement
ORDER BY SUM(population) DESC**

1. **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:

**Lorsque deux entités du MCD sont dans une relation Père Fils (*Constellation* est le père), la clé primaire de la table résultante glisse en tant que clé étrangère dans la table du fils (Etoile).**

Etoile (**Rang**, nom, distance, magnitude, déclinaison*,# id\_constellation*)
Constellation (**id\_constellation**, nom\_const, code\_Bayer)

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



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



🖙 A Lisbonne: 

*Ecrivons les requêtes SQL qui permettent d´afficher:*

 2) Le contenu de chacune des deux tables:

**SELECT \* FROM Etoile
SELECT \* FROM Constellation**

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

**SELECT nom\_const, id\_constellation FROM Constellation WHERE nom\_const='Orion'**

4) Le nom des étoiles de la constellation d´Orion enregistrées dans la base:
***On connaît maintenant l´id d´Orion (6)...*SELECT nom FROM Etoile WHERE id\_constellation=6**

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

**SELECT nom, declinaison FROM Etoile WHERE declinaison >38.7-90**

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

**SELECT nom, declinaison, id\_constellation FROM Etoile WHERE declinaison >90-38.7
*On aimerait connaître directement la constellation qui abrite ces deux étoiles. Nous le feront avec les jointures en question 9).***

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

7) La jointure complète des deux tables

**SELECT \* FROM Etoile
INNER JOIN Constellation
ON Etoile.id\_constellation = Constellation.id\_constellation**

***La clause INNER est facultative. Attention à bien utiliser la notation pointée lors d´une jointure.***

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

**SELECT Etoile.nom, Constellation.nom\_const FROM Etoile
INNER JOIN Constellation
ON Etoile.id\_constellation = Constellation.id\_constellation**

***La notation pointée n´est pas nécessaire en première ligne.***

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

**SELECT Etoile.nom, Constellation.nom\_const FROM Etoile
INNER JOIN Constellation
ON Etoile.id\_constellation = Constellation.id\_constellation
WHERE declinaison >90-38.7**

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

**SELECT COUNT(Etoile.nom), Constellation.nom\_const FROM Etoile
INNER JOIN Constellation
ON Etoile.id\_constellation = Constellation.id\_constellation
GROUP BY nom\_const
ORDER BY COUNT(Etoile.nom) DESC**