Bases de données - Langage SQL - TP n°2 -Enoncé

Base de données

On utilise dans ce TP la base de données "world" (accessible sur le web).
La base de données contient de très nombreuses tables, trois documents présentent les caractéristiques de la base de données :

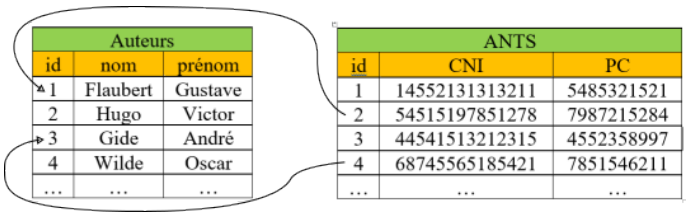
- Diagramme E-R (Entity-Relation) : tables (rectangles roses), attributs (ovales bleus) et tables de liaisons (losanges verts) qui permettent de relier plusieurs tables entre elles.
- RS (Relational schema) : définition des attributs des tables.
- Referential Dependencies (graphe des clés).

🛠 **Utiliser ces documents pour déterminer les informations pertinentes non fournies dans la formulation des requêtes (par exemple, attributs d'une table).**

Vocabulaire

📚 ***Types de dépendances entre tables***

- **Relation un-à-un (one-to-one) notée 1-1** : un enregistrement d'une table A est relié à un unique enregistrement d'une autre table B via un couple clé primaire A / clé étrangère unique dans B (une clé étrangère unique est une clé qui ne peut apparaître que dans un seul enregistrement de la table B).
  Exemple : une personne (table A) ne peut avoir qu'une seule carte d'identité (table B).
  Réalisation pratique en SQL :
  table_A AS *A* JOIN *table_B* AS *B* ON *A.cléPrimaire=B.cléEtrangère* (celle-ci étant unique).

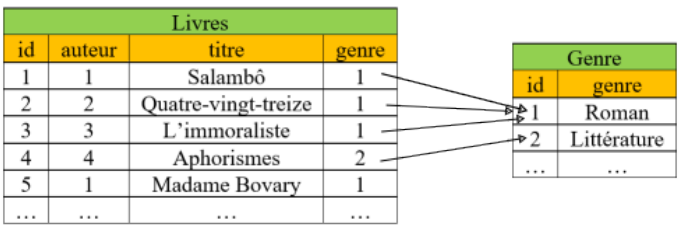


Auteurs		
id	nom	prénom
1	Flaubert	Gustave
2	Hugo	Victor
3	Gide	André
4	Wilde	Oscar
…	…	…

ANTS		
id	CNI	PC
1	14552131313211	5485321521
2	54515197851278	7987215284
3	44541513212315	4552358997
4	68745565185421	7851546211
…	…	…

FIGURE 1 : one-to-one

- **Relation un-à-plusieurs (one-to-many) notée 1-*** : un enregistrement d'une table A est relié à un ou plusieurs enregistrements d'une autre table B via un couple clé primaire A / clé étrangère non unique dans B (la clé étrangère peut apparaître dans plusieurs enregistrements de la table B).
  Exemple : un auteur de roman (table A) peut écrire plusieurs livres (table B) mais un livre n'est associé qu'à un seul auteur.
  Réalisation pratique en SQL :
  table_A AS *A* JOIN *table_B* AS *B* ON *A.cléPrimaire=B.cléEtrangère* (celle-ci n'étant pas nécessairement unique).

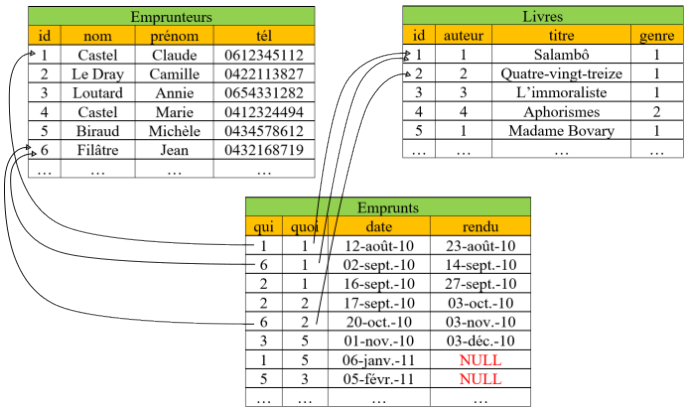


Livres			
id	auteur	titre	genre
1	1	Salambô	1
2	2	Quatre-vingt-treize	1
3	3	L'immoraliste	1
4	4	Aphorismes	2
5	1	Madame Bovary	1
…	…	…	…

Genre	
id	genre
1	Roman
2	Littérature
…	…

FIGURE 2 : many-to-one

- **Relation plusieurs-à-plusieurs (many-to-many) notée *-*** : deux tables A et B sont reliées via une troisième table appelée table de liaison ou d'association contenant deux clés étrangères reliées respectivement à une clé primaire dans chacune des tables A et B.
  Exemple : un rédacteur de presse (table A) peut écrire plusieurs articles (table B) et par ailleurs un même article (table B) peut avoir plusieurs rédacteurs (table A).
  Dans le TP n°1, la table Emprunts avec ses clés Qui et Quoi était une table de liaison entre la table Emprunteurs et la table Livres.
  Dans ce TP, la table encompasses est également une table de liaison.
  Ces tables sont les losanges verts dans le schéma E-R ci-dessus.
  Réalisation pratique en SQL :**double jointure**
  table_A JOIN *table_liaison* ON ... JOIN *table_B* ON ....



Emprunteurs

id	nom	prénom	tél
1	Castel	Claude	0612345112
2	Le Dray	Camille	0422113827
3	Loutard	Annie	0654331282
4	Castel	Marie	0412324494
5	Biraud	Michèle	0434578612
6	Filâtre	Jean	0432168719
…	…	…	…

Livres

id	auteur	titre	genre
1	1	Salambô	1
2	2	Quatre-vingt-treize	1
3	3	L'immoraliste	1
4	4	Aphorismes	2
5	1	Madame Bovary	1
…	…	…	…

Emprunts

qui	quoi	date	rendu
1	1	12-août-10	23-août-10
6	1	02-sept.-10	14-sept.-10
2	1	16-sept.-10	27-sept.-10
2	2	17-sept.-10	03-oct.-10
6	2	20-oct.-10	03-nov.-10
3	5	01-nov.-10	03-déc.-10
1	5	06-janv.-11	NULL
5	3	05-févr.-11	NULL
…	…	…	…

FIGURE 3 : many-to-many

1 Opérateur ensembliste UNION

✏ Requête n°1.a
Afficher la table *Continent* triée par ordre alphabétique des noms de continents.
✏ Requête n°1.b
Quelle requête doit-on écrire si on souhaite n'afficher que les n-uplets (ou enregistrements) concernant l'Afrique et l'Asie ?

> 🗄 ***UNION*** SELECT *attributs* FROM *nom_table_1* WHERE *formule_1*
> UNION
> SELECT *attributs* FROM *nom_table_2* WHERE *formule_2*
>
> Remarque : UNION est plus générale que OR car elle permet de rassembler deux requêtes distinctes (qui doivent cependant renvoyer le même nombre de colonnes avec les mêmes types d'attributs et dans le même ordre).

✏ Requête n°1.c
Retrouver les résultats précédents à l'aide d'une requête utilisant UNION.

2 Fonctions d'agrégation derrière SELECT

2.1 Moyenne d'un attribut

✏ Requête n°2
Quelle est la moyenne des surfaces des continents ? On utlisera le renommage *Surface_moyenne* pour cette colonne.

2.2 Recherche de renseignements associés à un extremum

La recherche du maximum d'un attribut est facile.

💻 👁 Par exemple, pour déterminer la surface du continent le plus vaste :

```
SELECT MAX(Area) FROM Continent
```

💻 👁 Trouver le continent le continent le plus vaste est également aisé : il suffit d'ajouter l'attribut correspondant dans les projections.

```
SELECT name, MAX(area) FROM continent
```

💡 La recherche de renseignements associée à la fonction d'agrégation est parfois plus compliquée.
La méthode la plus générale est illustrée avec ce même exemple (continent le plus vaste).

> ⚠ ***Erreur fréquente***
>
> **ATTENTION : la requête "intuitive" suivante est incorrecte**
> SELECT *attribut_1* FROM *table* WHERE *attribut_2* = MAX(*attribut_2*)
> ⇒ message d'erreur : Error : misuse of aggregate function MAX().
> **La fonction d'agrégation doit obligatoirement se situer après l'instruction SELECT (avant FROM) ou bien après HAVING.**

✏ Requête n°3
Quel est le continent le plus vaste ?

💡 ***Méthode générale***
Etape 1
Écrire la requête recherchant l'aire maximum (sans autre renseignement).
Etape 2
Il suffit de demander le nom dans la table Continent pour lequel Area est égale au résultat de la requête précédente (qui devient donc une sous-requête derrière WHERE)
Écrire la requête complète.

Conclusion
Dans le cas de l'égalité entre un attribut est sa valeur maximum, cette syntaxe est inutile comme précisé ci-dessus mais elle devient indispensable dans le cas d'une inégalité par exemple.
La requête suivante repose sur ce schéma :
`SELECT ... FROM ... WHERE ... > (SELECT AVG(...) FROM ...)`

✏ Requête n°4
Quels sont les continents dont la surface est supérieure à la surface moyenne des continents ?

3 Filtrage du nombre de résultats - LIMIT

LIMIT
SELECT ... FROM ... LIMIT *entier_1*, *entier_2*

✏ Requête n°5.a
- Afficher la table *Country* et constater qu'elle est longue...
- Afficher les 10 premiers n-uplets.

✏ Requête n°5.b
Afficher uniquement les n-uplets de "North Macedonia" à "Andorra" de la table Country sans utiliser OFFSET.

✏ Requête n°5.c
Afficher uniquement les n-uplets de "North Macedonia" à "Andorra" de la table Country en utilisant OFFSET.

✏ Requête n°5.d
Quels sont les trois continents de plus grande surface (classés par ordre décroissant de la surface) ?

✏ Requête n°5.e
Quel est le plus petit continent en termes de surface ?

4 Calculs mathématiques sur les données

Attribut sous forme de formule

Il est possible de créer un nouvel attribut en utilisant une formule à partir de plusieurs attributs d'une table.

SELECT *formule entre attributs* FROM ...

La formule est construite à partir des opérateurs + - * / et des fonctions mathématiques usuelles (**POWER**(x, n), **ROUND**(x, n), ...).

✏ Requête n°6 (préparatoire à la requête n°7)
Afficher la liste des cinq pays d'Afrique les plus peuplés ainsi que leur population classés par ordre décroissant de la population.

✏ Requête n°7
Afficher la liste des pays d'Afrique ainsi que leur **densité de population** (habitants / km^2) classés par ordre décroissant de la densité de population, en se limitant à une liste de cinq pays (on pourra utiliser l'alias *Densité* pour la colonne correspondante).

✏ Requête n°8.a
Quel est le pays d'Afrique le plus densément peuplé et quelle est sa densité de population ? On pourra utiliser l'alias *Densité_max* pour la colonne correspondante.

✏ Requête n°8.b
Quels sont les pays d'Afrique dont la densité de population est supérieure à la densité moyenne de ce continent ?

⚠ Remarque 1 - En cas de renommage, ne pas utiliser dans la sous-requête un alias créé dans la requête principale car **la sous-requête est évaluée avant la requête principale** donc un alias créé dans la requête principale est inconnu au moment de l'évaluation de la sous-requête.

Dans la requête ci-dessous l'alias "densité" est utilisé uniquement dans la requête principale, la sous-requête doit utiliser la formule "Population/Area".

```
SELECT Name, Population/Area AS densité
FROM Country AS c JOIN encompasses AS e ON c.code=e.country
WHERE continent='Africa' AND densité >
(SELECT AVG(Population/Area) FROM Country AS c JOIN encompasses AS e ON c.code=e.country)
```

⚠ Remarque 2 - On pourrait être tenté de n'utiliser qu'une seule table en remplaçant la sous-requête par GROUP BY ... HAVING mais cette solution ne fonctionne pas car dans ce cas ***AVG() est calculé pour chaque regroupement et non pour toute la table***.

La requête ci-dessous illustre cette mauvaise utilisation de HAVING et AVG.

L'utilisation correcte de HAVING suivi de AVG fait l'objet du paragraphe suivant.

```
SELECT Name, Population/Area AS densité
FROM Country AS c JOIN encompasses AS e ON c.code=e.country
WHERE continent='Africa'
GROUP BY Name
HAVING densité >= AVG(densité)
```

⚠ Remarque 3 - Lorqu'aucune fonction de regroupement n'est nécessaire, la syntaxe est très simple et ne nécessite pas de sous-requête (cf. requête ci-dessous).

✏ Requête n°8.c
Quels sont les pays d'Afrique dont la densité de population est supérieure ou égale à 1000 habitants par km^2 ?

5 Fonctions d'agrégation dans les regroupements derrière HAVING

🛠 Il est courant d'effectuer des calculs (somme, moyenne, max...) sur des **regroupements d'enregistrements** (ci-dessous, par continent par exemple).
Dans un tel cas, la fonction d'agrégation doit être placée dans la clause HAVING suivant l'instruction de regroupement GROUP BY.

✏ Requête n°9.a
Quels sont les continents dont la densité moyenne de population est supérieure à 50 hab/km^2 ?
Utiliser la fonction **ROUND**(x, n) afin d'obtenir des valeurs entières.

💡 Si on souhaite afficher le résultat obtenu via les fonctions d'agrégation (donc derrière SELECT), on peut utiliser un alias pour ce résultat et c'est cet alias qui sera utilisé derrière HAVING (afin de ne pas dupliquer le même code à deux endroits différents de la requête).

✏ Requête n°9.b
Quels sont les continents dont la densité moyenne de population est supérieure à 50 hab/km^2 ? Afficher également la densité.

Conclusion : sous-requête ou non ? Différences entre les requêtes n°8.b et 9.a (ou 9.b)

Dans la requête n°9.a, on sélectionne (HAVING) des enregistrements après avoir effectué un calcul (densité moyenne) nécessitant un regroupement (GROUP BY) ⇒ aucune sous-requête nécessaire (même calcul pour tous les regroupements).
Dans la requête n°8.b, on souhaite comparer une valeur (densité pour un continent) à une valeur nécessitant un calcul préalable (moyenne des densités) ⇒ sous-requête indispensable pour effectuer ce calcul préalable.

6 Fonctions d'agrégation - Usages incorrects

> ⚠ ***Erreurs fréquentes***
>
> Cf. deux requêtes ci-dessous.

On souhaite déterminer le pays dans lequel le nombre de langues parlées est le plus grand.

⚠ Exemple 1 - tentative "intuitive" la plus simple : MAX(COUNT(*)).
Erreur car les deux opérations (comptage et détermination de la valeur maximale) doivent être effectuées séparément (en deux temps).

```
SELECT country.name AS Pays, MAX(COUNT(*)) AS Langues_parlées
FROM country JOIN language ON country.code=language.country
GROUP BY Pays
```

Error : misuse of aggregate function COUNT()

⚠ Exemple 2 - Tentative de contournement de l'erreur précédente.
Erreur pour la même raison que ci-dessus, de plus l'usage de fonctions d'aggrégation est incorrect derrière WHERE.

```
SELECT country.name AS Pays, COUNT(*) AS Langues_parlées
FROM country JOIN language ON country.code=language.country
WHERE Langues_parlées = MAX(Langues_parlées)
GROUP BY Pays
```

Error : misuse of aliased aggregate Langues_parlées

Requête correcte : utilisation d'une sous-requête.

```
SELECT Pays, MAX(Langues_parlées)
FROM
(SELECT country.name AS Pays, COUNT(*) AS Langues_parlées
FROM country JOIN language ON country.code=language.country
GROUP BY Pays)
```

7 Utilisation de deux tables derrière FROM

Exécuter la requête ci-dessous calculant la fraction surfacique de chaque continent (par rapport à la surface de la totalité des continents) et observer le résultat. Une sous-requête est utilisée pour calculer la surface totale.

Bien noter que deux tables sont sélectionnées derrière le premier FROM : pour quelle raison ? (Que se passe-t-il si on supprime la table *Continent* ?)

```
SELECT name, Area / Surface_totale
FROM Continent, (SELECT SUM(Area) AS Surface_totale FROM Continent)
```

Si on omet la table *Continent*, une erreur est levée car la table générée par la sous-requête ne contient pas l'attribut *name*.

```
SELECT name, Area / Surface_totale
FROM (SELECT SUM(Area) AS Surface_totale FROM Continent)
```

Error : no such column : name

⚠ On pourrait être tenté de "simplifier" la syntaxe (cf. requête ci-dessous) mais cela ne fonctionne pas car la fonction d'agrégation SUM "écrase" tous les enregistrements : un seul calcul est effectué, pour le premier enregistrement (le calcul est faux pour les autres continents).

```
SELECT 100 * Area / SUM(Area)
FROM Continent
```

✏ Requête n°10
On souhaite répondre à la question suivante : "Quel est le pourcentage de la surface de l'Europe par rapport à l'ensemble des continents ?" (Question analogue posée à CCINP 2024)

✏ Requête n°11
Ecrire la requête permettant de calculer la somme des pourcentages de la table obtenue à la requête n°10.

🗄 ***FROM table_1, résultat_sous_requête_sur_table_1***
Il est possible deux sélectionner deux tables (noms des tables séparés par une virgule), dont la seconde est le résultat d'une sous-requête sur la première, derrière FROM lorsqu'on a besoin de faire référence aux attributs de la table initiale et aux mêmes attributs dans la table générée par la sous-requête (requêtes 10 et 11).

Remarque
La requête n°8.b peut être écrite selon cette technique mais la syntaxe obtenue n'est pas plus simple que la syntaxe initiale, cf. requête ci-dessous.

```
SELECT Name, Population/Area AS densité
FROM Country AS c JOIN encompasses AS e ON c.code=e.country,
(SELECT AVG(Population/Area) AS Moyenne FROM Country AS c JOIN encompasses AS e ON c.code=e.country)
WHERE continent='Africa' AND densité > Moyenne
```

8 Exercices

✏ Exercice n°1
Dans combien de pays la langue française est-elle parlée ?

✏ Exercice n°2
Quelles sont les langues parlées par moins de 10 000 personnes dans le monde (en tenant compte des pourcentages) ? Afficher également le nombre de locuteurs.

✏ Exercice n°3
Afficher le nombre de pays dans le monde ainsi que la population mondiale.

✏ Exercice n°4
Combien y a-t-il de villes traversées par une rivière dans chaque pays ? Classer sur le nombre.

✏ Exercice n°5
Quels sont les pays possèdant plus de 20 villes traversées par une rivière ? Classer par ordre alphabétique.

Chaînes et dates

Il existe des fonctions SQL dédiées au traitement de dates mais elles ne figurent pas au programme de CPGE et elles ne sont pas implémentées dans ce notebook.
Il est néanmoins possible d'effectuer des opérations simples car les dates de cette base de données sont de type chaîne de caractères.

SUBSTRING (*chaîne, a, b*) permet d'extraire la sous-chaîne débutant à l'incice a (la numérotation débute à **1**) et terminant à l'indice b inclus.

CAST (*chaîne* AS UNSIGNED) permet de convertir une chaîne en entier positif.

Exercice n°6
Déterminer les organisations internationales (table "organization") créées après 2010, ainsi que leur date de création.

Variante comparant deux chaînes et non deux nombres :

Exercice n°7
Quelle est l'année de création de l'OTAN (NATO en anglais) ?

Exercice n°8
Combien d'organisations comportent 'Europe' dans leur nom ?

Exercice n°9
Lister le nombre de créations d'organisations internationales par année.

Variante comparant des chaînes et non des nombres :