Interroger une base de données en langage SQL

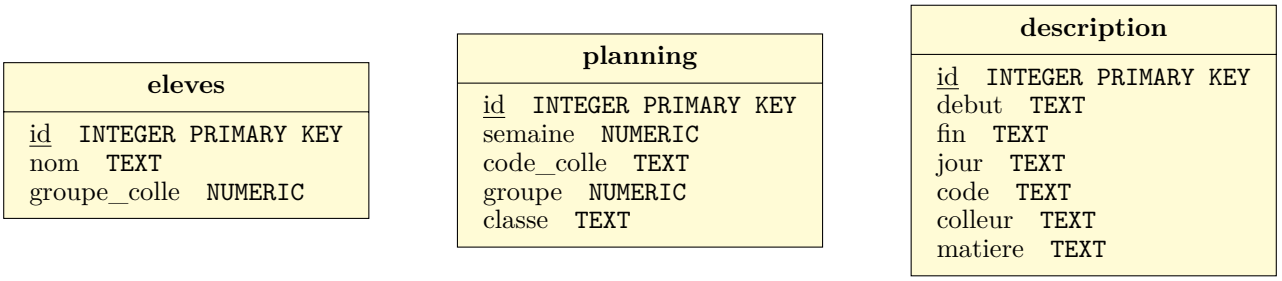
Préliminaires

Récupérer le dossier `TD11`.
L'enregistrer dans son dossier personnel.
Ce dossier contient

`colles.sqlite`	une BDD du colloscope d'une classe (PTSI)
`banque.sqlite`	une BDD banque
`SqlitemanPortable`	le dossier contenant le programme `SqlitemanPortable.exe` à utiliser

Lancer le programme `sqliteman` et ouvrir la base de données `colles.sqlite`

Première partie
Base de données du colloscope PTSI



Requêtes sur une seule table

1. Donner le nom et prénom de tous les élèves.
2. Donner le nom et prénom de tous les élèves classés par ordre alphabétique des prénoms.
3. Donner les codes de plages de colles.
   Ces codes constituent-ils une clé primaire pour la table considérée ?
4. Donner les numéros de groupe de colle et le nombre d'élèves pour chaque groupe.
   Utiliser `COUNT()`
5. Donner les groupes de 2 élèves (sans passer par une table intermédiaire).
6. Donner les noms, jours, plages des colleurs qui collent au même moment, on regroupera le résultat par jour.

Requêtes sur plusieurs tables

7. Donner une table avec nom complet des colleurs et les semaines pour le groupe 1.
8. Donner une table avec nom complet des colleurs pour la semaine 25 du groupe 1.
9. Donner une table avec nom de l'élève, colleur (nom complet) pour la semaine 25.
10. Donner une table donnant les semaines, les colleurs (nom complet), les jours, les plages horaires pour les semaines 23 à 28 vous concernant.

Deuxième partie
Base de données d'une banque

La base a été générée automatiquement grâce au site `http://www.generatedata.com/`

C'est un TP repris d'un polycopié de Serge Abiteboul, Benjamin Nguyen, Yannick Le Bras.
Ouvrir la base de données `banque.sqlite`

client
idclient INTEGER PRIMARY KEY
nom TEXT
prenom TEXT
ville TEXT

compte
idcompte INTEGER PRIMARY KEY
idproprietaire INTEGER
type TEXT

operation
idop INTEGER PRIMARY KEY
idcompte INTEGER
montant FLOAT
informations TEXT

Requête simple

11. Donnez les noms et prénoms de tous les clients de la base habitant à "Paris".

12. Donnez les identifiants de tous les comptes de type "Livret A"

13. Donnez les idop de type débit sur le compte d'idcompte 1

14. Donnez les idproprietaire des personnes possédant un Livret A

15. Donnez les idproprietaire des personnes ne possédant aucun Livret A.

16. Donnez le solde (= somme de toutes les opérations effectuées) de chaque compte

17. Donnez le solde de chaque compte, uniquement pour les comptes inférieurs ou égaux à 500 euros.

18. Donnez le solde des comptes d'identifiant parmi 1, 2, 5 qui ont un compte d'un montant inférieur à 500 euros.

19. Calculez le nombre moyen de comptes possédé par un client.

20. Affichez pour chaque opération son montant en dollars, en supposant que le taux de conversion soit une constante valant 1,3. Nommez cette colonne montantUSD

Un peu plus difficile ?

21. Donnez, pour chaque personne définie par son couple nom, prenom, la liste de ses comptes (identifiés par leur idcompte).
INDICATION On voit ici qu'on a besoin d'afficher des informations présentes dans deux tables : client et compte. Il faut identifier également le champ (=attribut) qui va permettre de faire le lien (jointure) entre une ligne de la première table et une ligne de la deuxième.

22. Donnez la liste des idcompte qui ont pour propriétaire une personne dont le prénom est "Marie".

23. Donnez la liste des opérations (montant et informations) sur chaque compte de Sylvie Moulin.

24. Donnez le noms et prénoms des clients ayant fait au moins une opération de débit sur l'un de leurs comptes.
INDICATION Les informations recherchées se trouvent dans la table operation, et les valeurs à tester dans la tableclient. Il faut de plus réussir à relier ces informations. Ce n'est possible qu'en utilisant la table compte.

25. Donnez pour chaque client identifié par son nom et prenom le nombre de comptes qu'il possède.
INDICATION Il s'agit ici de compter le nomdre d'idcompte distinct associé à chaque client.

26. Donnez pour chaque type de compte la somme totale disponible. Que faire si un type n'est associé à aucune opération ?

Difficile

27. Donnez pour chaque idcompte la valeur maximale de dépôt et de débit. S'il n'y a aucune valeur de dépôt ou de débit, alors le champ devra être `NULL`.

Exercices mettant en jeu une table ou requête imbriquée pour un calcul annexe

28. Donnez l'idproprietaire possédant le compte avec le plus grand numéro d'idcompte.
REMARQUE Il faut que la requête fonctionne quel que soit le contenu de la base !

29. Donnez les idproprietaire des personnes possédant moins de comptes que la moyenne calculée sur l'ensemble de la base de données.
INDICATION Comme la valeur qu'on veut tester (le nombre de comptes) est un agrégat, il faudra utiliser la clause HAVING.

Méthodologie pour rédiger une requête sur une BDD

```
SELECT [DISTINCT]..., [COUNT/SUM...]
FROM ... WHERE...
[GROUP BY... HAVING ...]
[ORDER BY...]
```

tests `AND, OR, IN` (*sous-requête*) *qui peuvent s'enchaîner*, `NOT, <>, =...`
fonctions `COUNT(), SUM(), MAX(), MIN(), AVG()`
joker *, typiquement `SELECT *`, `COUNT(*)` ou `COUNT(DISTINCT *)`
Nous proposons une méthodologie simple pour répondre à une requête basique exprimée « en français ».
Comment remplir une requête de type `SELECT ...FROM ...WHERE...` ?

1) Construction de la clause `SELECT`

a) Identifier les champs (= attributs) qui doivent être affichés, et les tables les contenant.

b) Mettre ces tables dans la clause `FROM` et éventuellement donner un nom de variable à chaque table (un alias)

c) Mettre les champs à afficher, si nécessaire préfixés par le nom de variable de la table correspondante dans la clause `SELECT`

d) (si nécessaire) Identifier les fonctions (`SUM`, `COUNT`...) à calculer, et les attributs nécessaires au calcul de ces fonctions, puis si nécessaire, ajouter des tables supplémentaires dans la clause `FROM`, et enfin ajouter la fonction dans la clause `SELECT`

2) Construction des groupes (optionnel)

a) Identifier les champs utilisés pour le partitionnement en groupes et les mettre dans la clause `GROUP BY`.

b) Compléter si nécessaire les tables de la clause `FROM`.

c) Compléter la clause `SELECT` en rajoutant les champs utilisés pour le partitionnement. En général, ces champs auront déjà été identifiés car ils vont a priori apparaître dans le résultat final, puisqu'ils s'agissent de critères de regroupement.

EXEMPLE une requête qui calcule le revenu moyen en fonction de la ville affichera forcément le nom de la ville de chaque groupe.

3) Construction des restrictions (`WHERE` et `HAVING`)

a) Regarder sur quels champs (ou groupes de champs) de quelles tables portent les restrictions.

b) Rajouter ces tables, si elles ne sont pas encore présentes, dans la clause `FROM`.

c) Rajouter les restrictions sur les champs en question dans la clause `WHERE`.

REMARQUE IMPORTANTE une condition de restriction peut tout à fait s'exprimer en utilisant une *sous-requête*.

Par exemple :

```
SELECT nom FROM tb_eleve WHERE villenaissance NOT IN (SELECT ville FROM tb_villes WHERE region <> 'normandie')
```

d) Rajouter les restrictions qui portent sur des valeurs agrégées (`GROUP`) dans la clause `HAVING`.

4) Construction des jointures

5) Rajouter la/les jointures avec `JOIN` et les *conditions de jointure* après `ON`, permettant de lier toutes les tables de la clause `FROM` les unes aux autres. Une telle jointure se fait en utilisant les *clés étrangères* des tables en question. Il peut arriver qu'il ne soit pas possible de lier une table aux autres. Cela signifie qu'il faudra introduire une ou plusieurs autres tables intermédiaires permettant de lier cette table par le biais d'un chemin composé de clés étrangères.

Remarque On pourrait se passer de `JOIN` et `ON` et n'utiliser que `WHERE` mais on perd en lisibilité et surtout en efficacité (voir la partie algèbre relationnelle).

6) Construction des sous-requêtes (optionnel)

Pour chaque sous requête utilisée dans la clause `WHERE`, vérifier qu'elle a besoin ou non d'être paramétrée. Une sous requête non paramétrée signifie que sa valeur ne dépend pas du moment où elle est calculée c'est-à-dire une sous requête peut être utilisée pour calculer le salaire moyen. Une sous requête paramétrée signifie que sa valeur dépend de la ligne qui est en train d'être traitée i.e. le fait qu'un idproprietaire donné possède un compte de type Livret A est une sous requête paramétrée. Dans le cas d'une sous requête paramétrée, celle-ci devra avoir une condition de paramétrisation dans la clause `WHERE`. Cette condition de paramétrisation fera intervenir un identifiant de table de la requête extérieure. (Voir exemples plus bas).

7) Présenter le résultat suivant un certain **classement**, **ORDER BY** (optionnel)

`A noter` On peut effectuer plusieurs requêtes (en appliquant la méthodologie précédente) puis les regrouper (`UNION`) ou en faire une différence (`EXCEPT`). On peut utiliser `UNION ALL` pour une réunion multi-ensembles (doublons possibles).

Rappel : écriture (moderne) des jointures

Jointure symétrique (entre deux tables)

```
SELECT co.type, op.montant FROM compte co, operation op
WHERE co.idcompte = op.idcompte GROUP BY co.type
```

peut s'écrire

```
SELECT co.type, op.montant FROM compte co JOIN operation op
ON co.idcompte = op.idcompte GROUP BY co.type
```

On peut enchaîner les jointures.