Deuxième TP de bases de données
ECG

Julien Reichert

2024/2025

2

Introduction

Dans ce TP, nous allons changer de base de données pour découvrir des requêtes plus avancées mentionnées comme non exigibles mais apparaissant au programme. Nous commencerons le travail sur la nouvelle base de données par une révision de la syntaxe de base.

Pour commencer, un exercice sans préparation des oraux d'HEC de 2024 et l'exercice de bases de données tombé à l'écrit d'Écricome la même année, légèrement moins bâclé que le sujet zéro paru en 2022 pour rester poli...

Exercice sans préparation Maths Appliquéees 6

On considère une base de donnée de location de voitures dont le schéma relationnel est donné par :

voiture (id_voiture, marque, modele, etat)
client (id_client, nom, prenom, adresse, num_permis)
location (id_loc, voiture, client, date_debut, date_fin)

1. Identifier les clés primaires et les éventuelles clés étrangères de chacune des tables.
2. L'état d'une voiture à louer est désigné par l'une de ces dénominations : « *neuf* », « *bon* », « *abimé* », « *au garage* ». Déterminer la liste des voitures qui sont actuellement dans l'un des deux meilleurs états.
3. Déterminer la liste des voitures (identifiant, marque et modèle) qui ont déjà été louées ou en cours de location.
4. Déterminer tous les clients (identifiant, nom et prénom) qui ont loué une Renault. On pourra réaliser une requête imbriquée, de la forme :

```
SELECT *
FROM table1
WHERE attribut IN (
    SELECT attribut2
    FROM table2
)
```

Partie IV

Un fabricant d'ordinateurs souhaite publier des données statistiques sur la durée de vie de ses appareils fabriqués à partir de l'an 2000. Dans une base de données, on dispose d'une table ordinateur contenant des informations sur tous les ordinateurs produits par le fabricant. Cette table possède les attributs (ou colonnes) suivants.

— `id` (de type `INTEGER`) : le numéro d'identification de l'ordinateur.

— `annee_fabrication` (de type `INTEGER`) : l'année de fabrication de l'ordinateur.

— `adresse_ip` (de type `INTEGER`) : l'adresse IP associée à l'ordinateur.

— `annee_panne` (de type `INTEGER`) : l'année où l'ordinateur a cessé de fonctionner, valant -1 si l'ordinateur est encore en état de marche.

Dans les questions qui suivent, en plus des commandes SQL au programme, on pourra utiliser les fonctions présentées dans l'**Annexe B** en fin de sujet.

9. (a) Écrire une requête SQL permettant de déterminer le nombre total d'ordinateurs produits par le fabricant.
   (b) Écrire une requête SQL permettant de déterminer le nombre d'ordinateurs ayant cessé de fonctionner exactement un an après leur production.
   (c) Dans cette question uniquement, on suppose que la durée de vie en années d'un ordinateur est une variable aléatoire de loi géométrique, de paramètre p inconnu.
   Expliquer de quelle manière le résultat des requêtes écrites dans les questions 9a et 9b peut être utilisé pour estimer le paramètre p.

10. Un attribut `duree_vie`, de type `INTEGER`, a été ajouté à la table `ordinateur`. Aux champs de l'attribut `duree_vie` a été affectée la valeur -1 .
   Écrire une requête SQL permettant de modifier la table `ordinateur` en affectant, pour chaque ordinateur, sa durée de vie à l'attribut `duree_vie`. Dans le cas des ordinateurs qui sont encore en état de marche, on ne modifiera pas la valeur -1 déjà affectée.

11. Dans cette question, on cherche à déterminer s'il est raisonnable de représenter la durée de vie d'un ordinateur par une variable aléatoire de loi géométrique d'un certain paramètre p que l'on cherchera à approcher.

   (a) Expliquer comment le résultat de la requête suivante permet d'obtenir une valeur approchée de p

```
SELECT AVG(duree_vie) FROM ordinateurs
```

   (b) La base de données compte au total 10000 ordinateurs. On exécute les requêtes suivantes :

```
SELECT COUNT(*)/10000 FROM ordinateurs WHERE duree_vie = 1 ;
SELECT COUNT(*)/10000 FROM ordinateurs WHERE duree_vie = 2 ;
   ⋮                    ⋮                 ⋮
SELECT COUNT(*)/10000 FROM ordinateurs WHERE duree_vie = 24 ;
```

   En utilisant les résultats de la question 8, expliquer de quelle manière les données de la table ordinateur peuvent être exploitées pour déterminer s'il est raisonnable de représenter la durée de vie d'un ordinateur par une variable aléatoire de loi géométrique.

Le TP

Les identifiants de connexion restent valides. De nouvelles tables indépendantes de celles utilisées pour le commerce en ligne ont été ajoutées entre temps.

Il s'agit de gérer des paris (sans enjeu !) sur les matchs de championnats sportifs comme la coupe du monde 2022. Les nouvelles tables sont les suivantes :

- `Equipes`, dont les attributs sont `Id_equipe` (entier) et `Pays` (chaîne) ;
- `Matchs`, dont les attributs sont `Id_match` (entier), `Date` (chaîne `"MM/JJ"`), `Phase` (chaîne de caractères pouvant valoir `"Groupe A"`, `"Quarts de finale"`, ...), `Equipe1` (entier) et `Equipe2` (entier) ;
- `Parieurs`, dont les attributs sont `Id_parieur` (entier) et `Nom_parieur` (chaîne) ;
- `Paris`, dont les attributs sont `Parieur` (entier), `Id_match` (entier), `Score1` (entier) et `Score2` (entier).
- `Resultats`, dont les attributs sont `Id_match` (entier), `Buts1` (entier) et `Buts2` (entier).

À partir de cette structure, on peut imaginer différentes manières de marquer des points sur les paris : en pronostiquant le bon score ou en se contentant d'annoncer le bon vainqueur (ou de deviner correctement qu'il y a un match nul).

Exercice 1 (révision) : Afficher la liste des parieurs.

Exercice 2 (révision) : Afficher les matchs où aucun but n'a été marqué.

Exercice 3 : Afficher les matchs où une équipe a marqué au moins trois buts.

Une possibilité est d'utiliser des opérateurs booléens dans la condition, mais en pratique pour des requêtes plus compliquées il existe aussi un opérateur pour la réunion (ainsi que pour l'intersection et pour la différence) sur des résultats de requêtes.

```
SELECT ...
UNION
SELECT ...
```

(Pour les autres opérations : `INTERSECT` et `EXCEPT`.)

Exercice 4 : Afficher les identifiants des équipes qui sont « receveuses » (`Equipe1`) pour les quarts de finale.

Exercice 5 : Même exercice mais en faisant apparaître le nom du pays à la place de l'identifiant et en renommant cet attribut `Pays` en `Receveur`.

Un attribut peut être renommé en utilisant le mot-clé `AS` suivi du nouveau nom. En pratique le mot-clé `AS` peut être omis mais c'est moins lisible.

```
SELECT ‘attribut‘ AS ‘alias‘ FROM ...
```

Par ailleurs, le renommage peut aussi concerner des tables. Ce sera nécessaire si une jointure fait intervenir deux fois la même table, causant une ambigüité qu'on ne peut pas lever autrement.

```
... ‘table‘ JOIN ‘meme_table‘ AS ‘autre_nom‘ ...
```

Exercice 6 : Afficher les quarts de finale avec les deux équipes, en renommant les attributs `Receveur` et `Visiteur`.

Exercice 7 : Afficher le nombre de paris effectués.

La fonction `COUNT` permet de récupérer le nombre de valeurs (sans éliminer les doublons) d'un attribut **en tant qu'attribut fictif, donc qui peut être mis après un** `SELECT`. Comme il y a pour chaque attribut une valeur par enregistrement (on exclut les valeurs `NULL`, hors-programme), n'importe quel attribut peut être compté, et en pratique on utilisera `COUNT(*)`. Attention, le moteur utilisé ici signale une erreur de syntaxe si la parenthèse est précédée d'une espace.

Un cas particulier : on écrit `COUNT(DISTINCT ‘attribut‘)` si on veut éliminer les doublons de l'attribut en question lors d'un comptage.

Exercice 8 : Afficher le nombre de personnes ayant fait au moins un pari.

Exercice 9 : Afficher le plus grand nombre de buts marqués par une équipe qui était `Equipe1`.

Cette fois-ci, la fonction à utiliser est `MAX`, sur un attribut là encore. D'autres fonctions sont au programme : `MIN`, `SUM` (somme) et `AVG` (moyenne).

Exercice 10 : Compter le nombre de points du parieur numéro 1 en comptant un point pour chaque pari où le score exact a été trouvé.

Exercice 11 : Même question mais en comptant un point pour chaque pari où le résultat (gagné / nul / perdu) est correct.

Pour faire le classement des parieurs, un regroupement est nécessaire. Ceci est hors programme mais pourra faire l'objet d'une démonstration en fin de séance. On en profitera pour illustrer `ORDER BY` pour trier les résultats affichés, et l'ensemble du programme sera traité.