Université de Carthage Institut Préparatoire aux Etudes d'Ingénieur de Nabeul Département : Mathématiques	IPEIN المعهد التحضيري للدراسات الهندسية بنابل Institut Préparatoire aux Etudes d'Ingénieurs de Nabeul	Année universitaire : 2022/2023 Filière : SM/SP/ST Niveau d'étude : 2ème année Semestre : 2 Nombre de pages : 2 pages Date : 02/03/2023, Durée : 1h30

DS Informatique N° 2 ~ Corrigé

> **ADHERENT** (NUMA, NOMA, PRENOMA, VILLEA, COT)
>
> **VOYAGE** (REFV, DESTV, DUREEV, COUTV, TYPEV)
>
> **DEPART** (#REFV, DATEDEP)
>
> **RESERVATION** (NDOSSIER, #NUMA, #REFV, #DATEDEP, ACOMPTE)

1. Décrire le rôle de la commande suivante écrite en Algèbre relationnel

$$\boldsymbol{R} = \boldsymbol{\pi}_{\text{REFV,DESTV,COUTV}}\left(\boldsymbol{\sigma}_{\text{DESTV='New York' et DUREEV>=3}}(\text{VOYAGE})\right)$$

> Afficher la référence, la destination et le coût des voyages organisés pour la destination "New York" et ayant une durée de 3 semaines ou plus.

2. Convertir, en Algèbre relationnel, la commande SQL suivante permettant d'afficher les noms et prénoms des adhérents ayant réservé un voyage pour la destination "Japon" avec un acompte de plus de 2000 DT.

> $$\boldsymbol{R1} = \boldsymbol{\pi}_{\text{REFV}}\left(\boldsymbol{\sigma}_{\text{REFV ='Japon'}}(\text{VOYAGE})\right)$$
>
> $$\boldsymbol{R2} = \boldsymbol{\sigma}_{\text{ACOMPTE>2000}}(\text{RESERVATION})$$
>
> $$\boldsymbol{R3} = \boldsymbol{\pi}_{\text{NUMA}}\left(\mathbf{R1} \bowtie \mathbf{R2}\right)$$
>
> $$\boldsymbol{R4} = \boldsymbol{\pi}_{\text{NOMA,PRENOMA}}\left(\text{ADHERENT} \bowtie \mathbf{R3}\right)$$

Répondre en langage **SQL** aux questions suivantes :

3. Combien d'adhérents ont réglé leur cotisation cette année ?

```
SELECT      COUNT(NUMA)
FROM        ADHERENT
WHERE       COTA = 1
```

4. Combien d'adhérents se sont inscrits à un voyage pour chaque type de voyage ?

```
SELECT      TYPEV, COUNT (DISTINCT NUMA)
FROM        RESERVATION R JOIN VOYAGE V ON R.REFV = V.REFV
GROUP BY    TYPEV
```

5. Que fait la requête SQL suivante et donner une autre version qui donne le même résultat.

```
SELECT      *
FROM        ADHERENT
WHERE       NUMA NOT IN (SELECT DISTINCT NUMA FROM RESERVATION)
```

Réponse : Cette requête SQL permet d'afficher les adhérents qui n'ont jamais voyagés.

```
SELECT  *
FROM    ADHERENT
WHERE   NUMA  IN  (SELECT NUMA FROM ADHERENT)
                  EXECPT
                  (SELECT DISTINCT NUMA FROM RESERVATION)
```

6. Afficher les types de voyage ayant plus de 5 voyages organisés.

```
SELECT   TYPEV, COUNT(*) AS nb_voyages
FROM     VOYAGE
GROUP BY TYPEV
HAVING   COUNT(*) > 5;
```

7. Mettre à jour le coût de tous les voyages qui durent plus de 3 semaines en ajoutant 10% au coût initial.

```
UPDATE VOYAGE
SET    COUTV = COUTV * 1.1
WHERE  DUREEV > 3
```

8. Retrouver le nombre de réservations, la somme et la moyenne des acomptes versés par voyage dont le nombre de réservation est supérieur ou égal à 2.

```
SELECT        r.refv, COUNT(*), SUM(r.acompte), AVG(r.acompte)
FROM          reservation r
GROUP BY      refv
HAVING        COUNT(*)>=2
```

9. Quel est le voyage qui a reçu le plus grand nombre de réservations pour la date de départ '2023-07-01' ?

```
SELECT REFV, COUNT(*) AS NB_RESERVATIONS
FROM RESERVATION
WHERE DATEDEP = '2023-07-01'
GROUP BY REFV
HAVING COUNT(*) = (SELECT MAX(NB_RESERVATIONS)
                   FROM (SELECT REFV, COUNT(*) AS NB_RESERVATIONS
                         FROM RESERVATION
                         WHERE DATEDEP = '2023-07-01'
                         GROUP BY REFV)
                  )
```

10. Quels sont les adhérents qui ont réservé tous les voyages proposés pour l'année prochaine ?

```
SELECT        NUMA, COUNT(DISTINCT REFV) AS NB_VOYAGES_RESERVES
FROM          RESERVATION
GROUP BY      NUMA
HAVING        COUNT(DISTINCT REFV) =
              (SELECT COUNT(*)
               FROM VOYAGE V JOIN DEPART D ON V.REFV = D.REFV
               WHERE YEAR(DATEDEP) = YEAR (Date())+1
               )
```

11. Quelle est la destination la plus populaire pour les adhérents qui ont réservé un voyage ?

```
SELECT        V.DESTV, COUNT(*) AS NB_RESERVATIONS
FROM          VOYAGE V
JOIN          DEPART D ON V.REFV = D.REFV
JOIN          RESERVATION R ON D.REFV = R.REFV
AND           D.DATEDEP = R.DATEDEP
GROUP BY      V.DESTV
ORDER BY      NB_RESERVATIONS DESC
LIMIT         1
```

12. Sélectionner les noms et prénoms des adhérents qui ont réservé un voyage pour la même date de départ qu'un autre adhérent.

```
SELECT a1.NOMA, a1.PRENOMA
FROM   ADHERENT a1, ADHERENT a2, RESERVATION r1, RESERVATION r2
WHERE  a1.NUMA = r1.NUMA
AND    a2.NUMA = r2.NUMA
AND    r1.DATEDEP = r2.DATEDEP
AND    r1.NDOSSIER <> r2.NDOSSIER
```

13. Quel est le montant total des acomptes versés par les adhérents pour chaque voyage ?

```
SELECT    REFV, SUM(ACOMPTE)
FROM      RESERVATION
GROUP BY  REFV
```

14. Quels sont les voyages pour lesquels le montant total des acomptes versés est inférieur à la moitié du coût total du voyage ?

```
SELECT REFV, DESTV
FROM VOYAGE
JOIN   (SELECT REFV, SUM(ACOMPTE) AS MONTANT
        FROM RESERVATION
        GROUP BY REFV) AS ACOMPTE_PAR_VOYAGE
ON VOYAGE.REFV = ACOMPTE_PAR_VOYAGE.REFV
WHERE MONTANT < COUTV / 2
```

15. Supprimer toutes les réservations des adhérents qui n'ont pas réglé leur cotisation de l'année en cours.

```
DELETE FROM RESERVATION
WHERE NUMA IN (SELECT NUMA FROM ADHERENT WHERE COTA = 0)
```

16. Ajouter une colonne nommée *NB_PLACES_DISPONIBLE* à la table VOYAGE pour suivre le nombre de places disponibles et réservées par voyage. Le nombre de places est initialisé à 55 et décrémente de 1 à chaque réservation.

```
ALTER TABLE VOYAGE
ADD COLUMN NB_PLACES_DISPONIBLE INT DEFAULT 55
```

17. Ajouter une nouvelle réservation pour l'adhérent numéro 202312 pour un voyage de référence JAP456, avec une date de départ le 1er mai 2023 et un acompte de 1000 DT.

```
INSERT INTO RESERVATION (NDOSSIER, NUMA, REFV, DATEDEP, ACOMPTE)
SELECT MAX(r.NDOSSIER) + 1, 202312, 'JAP456', '2023-05-01', 1000
FROM   RESERVATION r ;

UPDATE  VOYAGE
SET     NB_PLACES_DISPONIBLE = NB_PLACES_DISPONIBLE - 1
WHERE   REFV = 'JAP456'
```

18. Supprimer les réservations pour les voyages qui ont déjà eu lieu.

```
DELETE FROM RESERVATION  WHERE DATEDEP < DATE(NOW())
```

19. Donner la requête SQL permettant de créer la table **PAIEMENT**

```
CREATE TABLE PAIEMENT (
ID INT PRIMARY KEY NOT NULL AUTOINCREMENT,
RES INT NOT NULL,
DATEPAIEMENT DATE,
MONTANTPAIEMENT FLOAT,
FOREIGN KEY(RES) REFERENCES RESERVATION(NDOSSIER)
)
```

20. Créer un script Python utilisant SQLite permettant d'extraire des informations sur les adhérents ayant des paiements en retard pour un voyage donné, et les écrire dans un fichier texte.

```
import sqlite3 # Connexion à la base de données
conn = sqlite3.connect('club_voyages.db')
cur = conn.cursor()
# Requête SQL pour récupérer les informations des adhérents ayant des paiements
en retard pour un voyage donné
# Ici, on recherche les adhérents ayant des paiements en attente pour un voyage
ayant pour référence 1, et dont la date de départ est avant la date actuelle
sql_query = """
SELECT   a.NOMA, a.PRENOMA,
         (COUTV - ACOMPTE - SUM(p.MONTANTPAIEMENT)) as RESTE_A_PAYER
FROM     ADHERENT a
JOIN     RESERVATION r ON a.NUMA = r.NUMA
JOIN     PAIEMENT p ON r.NDOSSIER = p.RES
WHERE    r.REFV = ?
AND      DATEDIFF('day',r.DATEDEP, Date())<2
GROUP BY P.RES
HAVING   RESTE_A_PAYER < (SELECT COUTV - r.ACOMPTE
                          FROM VOYAGE V
                          WHERE V.REFV = r.REFV)
"""
# Demande à l'utilisateur de saisir la référence du voyage
ref_voyage = input("Entrez la référence du voyage : ")

# Exécution de la requête et récupération des résultats
cur.execute(sql_query, ref_voyage)
results = cur.fetchall()
# Écriture des résultats dans un fichier texte
with open('adhérents_en_retard.txt', 'w') as f:
       f.write('Adhérents ayant des paiements en retard pour le voyage :{}\n'
              .format(ref_voyage))
       for row in results:
              f.write("{}, {}, {} DT\n".format(row[0], row[1], row[2]))
# Fermeture de la connexion à la base de données
conn.close()
```