Solutions de l'examen de Bases de données (Session 2022–2023)

A. Requêtes en algèbre relationnelle et en SQL sans jointures ni agrégats

1. **Campings (IdC, Nom) dont la capacité est supérieure à 50 Algèbre relationnelle :**

$$\pi_{\text{IdC,Nom}}\big(\sigma_{\text{Capacite}>50}(\text{CAMP})\big).$$

Explication : On applique d'abord la sélection sur l'attribut `Capacite` de la relation `CAMP` pour ne retenir que les enregistrements supérieurs à 50, puis on projette les attributs `IdC` et `Nom`.
SQL :

```
SELECT IdC, Nom
FROM CAMP
WHERE Capacite > 50;
```

2. **Personnes (NSS, Nom) ayant fait une réservation de plus de 5 jours Algèbre relationnelle :**

$$\pi_{\text{NSS,Nom}}\Big(\sigma_{\text{NbJ}>5 \,\wedge\, P.NSS=G.NSS \,\wedge\, G.IdGP=R.IdGP}(\text{PERS} \times \text{GPE} \times \text{RESA})\Big).$$

Explication : On considère le produit cartésien de `PERS`, `GPE` et `RESA`, on restreint aux tuples où l'âge de séjour `NbJ` dépasse 5 et où les conditions d'appartenance (`PERS.NSS=GPE.NSS` et `GPE.IdGP=RESA.IdGP`) sont satisfaites, puis on projette les attributs `NSS` et `Nom`.
SQL :

```
SELECT DISTINCT P.NSS, P.Nom
FROM PERS P, GPE G, RESA R
WHERE P.NSS = G.NSS
  AND G.IdGP = R.IdGP
  AND R.NbJ > 5;
```

3. **Personnes (NSS, Nom) ayant campé au « Camping du Soleil » Algèbre relationnelle :**

$$\pi_{P.NSS,P.Nom}\Big(\sigma_{C.Nom='CampingduSoleil' \,\wedge\, R.IdCamp=C.IdC \,\wedge\, G.IdGP=R.IdGP \,\wedge\, P.NSS=G.NSS}(\text{PERS} \times \text{GPE} \times \text{RESA} \times \text{CAMP})\Big).$$

Explication : Le produit cartésien de `PERS`, `GPE`, `RESA` et `CAMP` est filtré pour ne retenir que les réservations liées au camping nommé « Camping du Soleil », puis on projette `NSS` et `Nom`.
SQL :

```
SELECT DISTINCT P.NSS, P.Nom
FROM PERS P, GPE G, RESA R, CAMP C
WHERE P.NSS = G.NSS
  AND G.IdGP = R.IdGP
  AND R.IdCamp = C.IdC
  AND C.Nom = 'Camping du Soleil';
```

4. **Personnes (NSS, Nom) ayant séjourné dans un même camping que « Zoé » Algèbre relationnelle :**

$$\pi_{P_2.NSS,P_2.Nom}\Big(\sigma_{P_1.NSS=G_1.NSS \wedge G_1.IdGP=R_1.IdGP \wedge G_1.NSS='Zo' \wedge P_2.NSS=G_2.NSS \wedge G_2.IdGP=R_2.IdGP \wedge R_1.IdCamp=R_2.IdCamp}(P_1 \times G_1 \times R_1 \times P_2 \times G_2 \times R_2)\Big).$$

Explication : On distingue deux occurrences des relations `PERS`, `GPE` et `RESA` : l'une pour « Zoé » (indices `1`) et l'autre pour les autres personnes (indices `2`). On impose que `Zoé` ait réservé dans un camping dont l'identifiant coïncide avec celui d'une réservation faite par une autre personne, et on projette les informations de cette dernière.
SQL :

```
SELECT DISTINCT P2.NSS, P2.Nom
FROM PERS P1, GPE G1, RESA R1,
     PERS P2, GPE G2, RESA R2
WHERE P1.NSS = G1.NSS
  AND G1.IdGP = R1.IdGP
  AND G1.NSS = 'Zoé'
  AND P2.NSS = G2.NSS
  AND G2.IdGP = R2.IdGP
  AND R1.IdCamp = R2.IdCamp;
```

5. **Personnes (NSS, Nom) n'ayant jamais campé Algèbre relationnelle :**

$$\pi_{\text{NSS,Nom}}(\text{PERS}) \; - \; \pi_{\text{NSS,Nom}}(\sigma_{P.NSS=G.NSS \wedge G.IdGP=R.IdGP}(\text{PERS} \times \text{GPE} \times \text{RESA})).$$

Explication : On prend la différence entre l'ensemble de toutes les personnes et celles ayant au moins une réservation (via `GPE` et `RESA`).
SQL :

```
SELECT NSS, Nom
FROM PERS
```

```
EXCEPT
SELECT DISTINCT P.NSS, P.Nom
FROM PERS P, GPE G, RESA R
WHERE P.NSS = G.NSS
  AND G.IdGP = R.IdGP;
```

6. **Personnes (NSS, Nom) n'ayant jamais fréquenté un même camping que « Zoé »**
Algèbre relationnelle :

$$\pi_{\text{NSS,Nom}}(\text{PERS}) \; - \; \pi_{P_2.NSS,P_2.Nom}(\text{résultat de la requête A.4}).$$

Explication : On soustrait à l'ensemble de toutes les personnes celles identifiées en A.4.
SQL :

```
SELECT NSS, Nom
FROM PERS
EXCEPT
SELECT DISTINCT P2.NSS, P2.Nom
FROM PERS P1, GPE G1, RESA R1,
     PERS P2, GPE G2, RESA R2
WHERE P1.NSS = G1.NSS
  AND G1.IdGP = R1.IdGP
  AND G1.NSS = 'Zoé'
  AND P2.NSS = G2.NSS
  AND G2.IdGP = R2.IdGP
  AND R1.IdCamp = R2.IdCamp;
```

7. **Personnes (NSS, Nom) ayant effectué au moins deux séjours dans le même camping**
Algèbre relationnelle :

$$\pi_{P.NSS,\,P.Nom}$$
$$(\sigma_{P.NSS=G_1.NSS \wedge G_1.IdGP=R_1.IdGP \wedge P.NSS=G_2.NSS \wedge G_2.IdGP=R_2.IdGP \wedge \quad R_1.IdCamp=R_2.IdCamp \wedge (R_1.Date \neq R_2.Date}$$
$$(P \times G_1 \times R_1 \times G_2 \times R_2)$$

Explication : On utilise deux copies de `GPE` et `RESA` pour détecter deux réservations distinctes (dates différentes) dans un même camping pour une même personne.
SQL :

```
SELECT DISTINCT P.NSS, P.Nom
FROM PERS P,
     GPE G1, RESA R1,
     GPE G2, RESA R2
WHERE P.NSS = G1.NSS
  AND G1.IdGP = R1.IdGP
  AND P.NSS = G2.NSS
```

```
  AND G2.IdGP = R2.IdGP
  AND R1.IdCamp = R2.IdCamp
  AND R1.Date <> R2.Date;
```

B. Requête en algèbre relationnelle uniquement

8. Personnes (NSS, Nom) ayant fréquenté *tous* les campings 3 étoiles de la ville de « Dax »

$$\text{Soit } C = \sigma_{\text{Ville}='Dax' \,\wedge\, \text{Etoile}=3}(\text{CAMP}),$$

$$\text{Fréq} = \pi_{\text{NSS,IdCamp}}(\text{PERS} \underset{\text{NSS=NSS}}{\bowtie} \text{GPE} \underset{\text{IdGP=IdGP}}{\bowtie} \text{RESA}),$$

$$\text{Résultat} = \pi_{\text{NSS,Nom}}\big((\text{Fréq} \div \pi_{\text{IdCamp}}(C)) \underset{\text{NSS=NSS}}{\bowtie} \text{PERS}\big).$$

Explication : On effectue une division relationnelle entre la relation des fréquents (mappings `NSS`–`IdCamp`) et l'ensemble des campings 3 étoiles de « Dax ». Le résultat donne les `NSS` ayant visité *tous* ces campings, puis on rejoint pour obtenir le nom.

C. Requêtes SQL avec agrégats, sous-requêtes et vues (sans jointures explicites)

9. **Nombre de réservations dans les campings de la ville de « Dax »**

```
SELECT COUNT(*) AS nb_reservations
FROM RESA R, CAMP C
WHERE R.IdCamp = C.IdC
  AND C.Ville = 'Dax';
```

10. **Villes dont le nombre de campings 3 étoiles est supérieur à la moyenne nationale**

```
SELECT Villes.Ville
FROM (
  SELECT Ville, COUNT(*) AS nb3
  FROM CAMP
  WHERE Etoile = 3
  GROUP BY Ville
) AS Villes
WHERE nb3 > (
  SELECT AVG(nb3)
  FROM (
```

```
    SELECT COUNT(*) AS nb3
    FROM CAMP
    WHERE Etoile = 3
    GROUP BY Ville
  ) AS Moyenne
);
```

11. **Personnes (NSS, Nom) ayant fréquenté tous les campings 3 étoiles de « Dax »**

```
SELECT P.NSS, P.Nom
FROM PERS P
WHERE NOT EXISTS (
  SELECT C.IdC
  FROM CAMP C
  WHERE C.Ville = 'Dax'
    AND C.Etoile = 3
    AND NOT EXISTS (
      SELECT *
      FROM GPE G, RESA R
      WHERE G.NSS = P.NSS
        AND G.IdGP = R.IdGP
        AND R.IdCamp = C.IdC
    )
);
```

D. Normalisation du schéma `CarLoc`

Le schéma initial :

$$CarLoc(\ IdCl, NomCl, IdCar, Type, DDeb, DFin, Cot, IdAg, NomAg\ )$$

avec l'ensemble des dépendances fonctionnelles F :

- $IdCl \rightarrow NomCl$
- $IdAg \rightarrow NomAg$
- $NomAg \rightarrow IdAg$
- $IdCl, IdCar \rightarrow DDeb, DFin$
- $IdCar \rightarrow Type, Cot, IdAg, NomAg$

Q1. Clés du schéma $(CarLoc, F)$

On cherche un ensemble minimal K tel que $K^+ = \{IdCl, NomCl, IdCar, Type, DDeb, DFin, Cot, IdAg, NomA$
Vérifions $\{IdCl, IdCar\}$:

$$\{IdCl, IdCar\}^+ \supseteq \{IdCl, IdCar\}$$
$$\xrightarrow{IdCl \to NomCl} \{IdCl, IdCar, NomCl\}$$
$$\xrightarrow{IdCar \to Type, Cot, IdAg, NomAg} \{IdCl, IdCar, NomCl, Type, Cot, IdAg, NomAg\}$$
$$\xrightarrow{IdCl, IdCar \to DDeb, DFin} \{IdCl, IdCar, NomCl, Type, Cot, IdAg, NomAg, DDeb, DFin\}.$$

Aucun sous-ensemble propre ne détermine ainsi toutes les attributs. **Donc la clé candidate est** $\{IdCl, IdCar\}$.

Q2. Vérification de la 3ème forme normale (3FN)

Une FD $X \to A$ viole la 3FN si X n'est pas une super-clé et A n'est pas un attribut prime (appartenant à une clé).

— $IdCl \to NomCl$: $IdCl$ n'est pas super-clé, $NomCl \notin \{IdCl, IdCar\} \to$ violation.
— $IdAg \to NomAg$: $IdAg$ n'est pas super-clé, $NomAg$ non prime $\to$ violation.
— $NomAg \to IdAg$: $NomAg$ n'est pas super-clé, $IdAg$ non prime $\to$ violation.

Conclusion : Le schéma n'est pas en 3ème forme normale.

Q3. Non conformité à la forme normale de Boyce–Codd (FNBC)

En FNBC, toute FD non triviale $X \to A$ doit avoir X super-clé. Or, par exemple, $IdCar \to Type$ avec $IdCar$ non super-clé $\to$ violation de BCNF. **Donc le schéma n'est pas en FNBC.**

Q4. Décomposition en FNBC

On applique l'algorithme de décomposition :

1. FD violant BCNF : $IdCar \to Type, Cot, IdAg, NomAg$. Décomposition en :
   — $R_1(IdCar, Type, Cot, IdAg, NomAg)$
   — $R_2(IdCl, NomCl, IdCar, DDeb, DFin)$
2. Vérification et seconde décomposition de R_1 : FD interne $NomAg \to IdAg$ violant BCNF dans R_1. On décompose R_1 en :
   — $R_{1a}(NomAg, IdAg)$
   — $R_{1b}(IdCar, Type, Cot, IdAg)$
3. Vérification et décomposition de R_2 : FD interne $IdCl \to NomCl$ violant BCNF dans R_2. On décompose R_2 en :
   — $R_{2a}(IdCl, NomCl)$

— $R_{2b}(IdCl, IdCar, DDeb, DFin)$

4. Vérification finale :
   — R_{1a} : $NomAg \to IdAg$ et $IdAg \to NomAg$ → les deux attributs sont clés → BCNF.
   — R_{1b} : $IdCar \to Type, Cot, IdAg \to IdCar$ clé de R_{1b} → BCNF.
   — R_{2a} : $IdCl \to NomCl$ avec $IdCl$ clé → BCNF.
   — R_{2b} : $IdCl, IdCar \to DDeb, DFin$ avec LHS clé → BCNF.

Schéma final en FNBC :

$$R_{1a}(NomAg, IdAg), \quad R_{1b}(IdCar, Type, Cot, IdAg),$$
$$R_{2a}(IdCl, NomCl), \quad R_{2b}(IdCl, IdCar, DDeb, DFin).$$