

**Conservatoire National des Arts et Métiers**

292 Rue St Martin 75141 Paris Cedex 03

**Informatique - CNAM, Paris**

**Bases de données relationnelles  
NFP 107 et NFP 107J**

(ancien SGBD B7, UV 19786 (HTO) et 21928 (ICPJ))

**Exercices dirigés**

B. Amann, V. Gouet-Brunet, D. Gross-Amblard, J. Lafaye, P. Rigaux,  
M. Scholl et D. Vodislav

**14 février 2006**



---

# Table des matières

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Conception</b>                                      | <b>3</b>  |
| 1.1      | Interprétation de schémas entité/association           | 3         |
| 1.1.1    | Centre médical   | 3         |
| 1.1.2    | Tournoi de tennis                                      | 4         |
| 1.1.3    | Un journal   | 4         |
| 1.2      | Modèle relationnel (rappel cycle A)                    | 4         |
| 1.3      | Rétro-conception                                       | 6         |
| <b>2</b> | <b>Algèbre Relationelle</b>                            | <b>7</b>  |
| 2.1      | Sélection et Projection                                | 7         |
| 2.2      | Jointure relationnelle                                 | 7         |
| 2.3      | Auto-Jointure et Renommage                             | 8         |
| <b>3</b> | <b>Algèbre - SQL : Employés - Départements</b>         | <b>9</b>  |
| 3.1      | Schéma   | 9         |
| 3.1.1    | Relation des Employés (EMP)                            | 9         |
| 3.1.2    | Relation des Départements (DEPT)                       | 9         |
| 3.2      | Opérations Algébriques                                 | 9         |
| 3.3      | Requêtes   | 10        |
| 3.3.1    | Interrogation d'une seule Relation                     | 10        |
| 3.3.2    | Jointures  | 10        |
| 3.3.3    | Valeurs Nulles, Tris, Groupes, Agrégats et Expressions | 11        |
| <b>4</b> | <b>Algèbre - SQL : Appartements - Écoles</b>           | <b>12</b> |
| 4.1      | Schéma   | 12        |
| 4.2      | Requêtes   | 13        |
| 4.3      | Mise à jour  | 14        |
| 4.4      | Contraintes  | 14        |
| <b>5</b> | <b>SQL - Algèbre : Cinémas - Films</b>                 | <b>15</b> |
| 5.1      | Schéma   | 15        |
| 5.2      | Requêtes   | 15        |
| 5.2.1    | Interrogation d'une seule relation                     | 15        |
| 5.2.2    | Jointures  | 15        |
| 5.2.3    | Difference   | 16        |
| 5.2.4    | Division   | 16        |
| <b>6</b> | <b>Organisation Physique</b>                           | <b>17</b> |
| <b>7</b> | <b>Algorithmes de Jointure</b>                         | <b>19</b> |
| <b>8</b> | <b>Optimisation de Requêtes</b>                        | <b>20</b> |

---

|          |   |           |
|----------|---|-----------|
| <b>9</b> | <b>Concurrence</b>  | <b>24</b> |
| 9.1      | Sérialisabilité et recouvrabilité . . . . .                     | 24        |
| 9.1.1    | Graphe de sérialisation et équivalence des exécutions . . . . . | 24        |
| 9.1.2    | Recouvrabilité . . . . .  | 24        |
| 9.2      | Contrôle de concurrence . . . . .                               | 24        |
| 9.2.1    | Verrouillage à 2 phases . . . . .                               | 24        |
| 9.2.2    | Verrouillage, interblocage . . . . .                            | 25        |
| 9.2.3    | Verrouillage hiérarchique . . . . .                             | 25        |
| 9.3      | Reprise après panne . . . . .                                   | 26        |
| 9.3.1    | Journalisation . . . . .  | 26        |
| 9.4      | Concurrence : Gestion Bancaire . . . . .                        | 26        |
| 9.5      | Concurrence : Agence environnementale . . . . .                 | 27        |

---

# Chapitre 1

## Conception

### 1.1 Interprétation de schémas entité/association

#### 1.1.1 Centre médical

On vous donne un schéma E/A (figure 1.1) représentant des visites dans un centre médical. Répondez aux questions suivantes **en fonction des caractéristiques de ce schéma** (i.e. : indiquez si la situation décrite est représentable, indépendamment de sa vraisemblance).

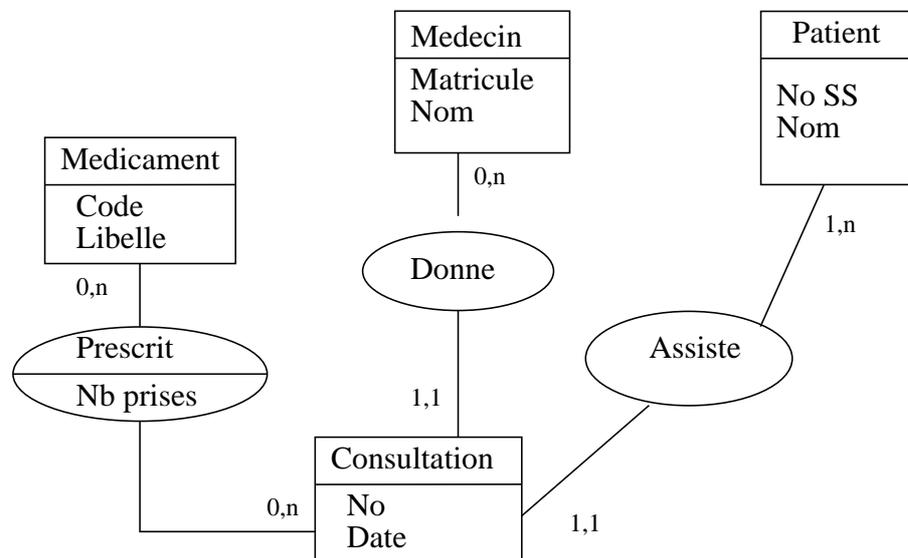


FIG. 1.1 – Centre médical

**Exercice A** : Un patient peut-il effectuer plusieurs visites ?

**Exercice B** : Un médecin peut-il recevoir plusieurs patients dans la même consultation ?

**Exercice C** : Peut-on prescrire plusieurs médicaments dans une même consultation ?

**Exercice D** : Deux médecins différents peuvent-ils prescrire le même médicament ?

### 1.1.2 Tournoi de tennis

Le second schéma (figure 1.2) représente des rencontres dans un tournoi de tennis.

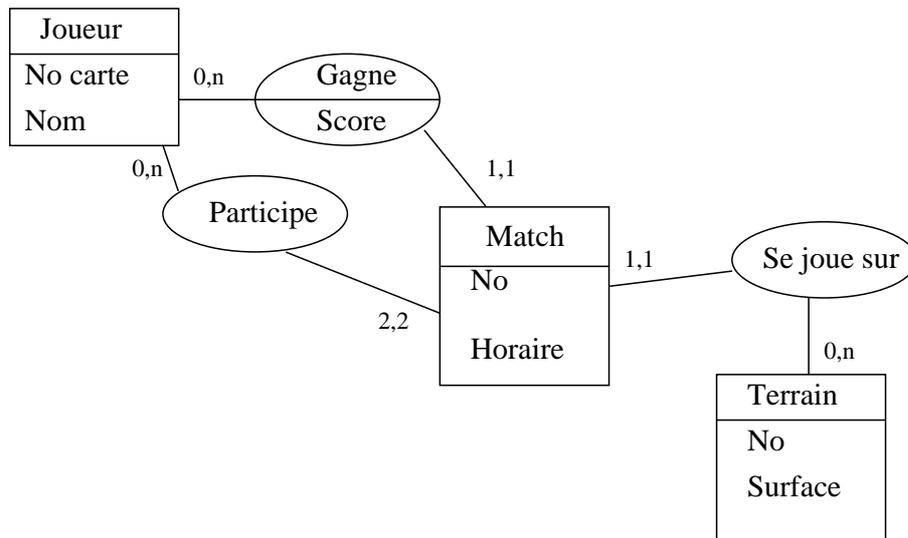


FIG. 1.2 – Tournoi de tennis

**Exercice A :** Peut-on jouer des matchs de double ?

**Exercice B :** Un joueur peut-il gagner un match sans y avoir participé ?

**Exercice C :** Peut-il y avoir deux matchs sur le même terrain à la même heure ?

### 1.1.3 Un journal

Pour vous entraîner : voici le schéma E/A (figure 1.3 du système d'information (très simplifié) d'un quotidien.

**Exercice A :** Un article peut-il être rédigé par plusieurs journalistes ?

**Exercice B :** Un article peut-il être publié plusieurs fois dans le même numéro ?

**Exercice C :** Peut-il y avoir plusieurs articles sur le même sujet dans le même numéro ?

## 1.2 Modèle relationnel (rappel cycle A)

**Exercice A :** Pour chacun des schémas E/A donnés précédemment, construire le schéma relationnel correspondant. Indiquez précisément :

- La clé primaire.
- Les clés étrangères.
- Les contraintes éventuelles.

**Exercice B :** Donnez la commande **Create Table** pour les tables *Consultation* et *Match*.

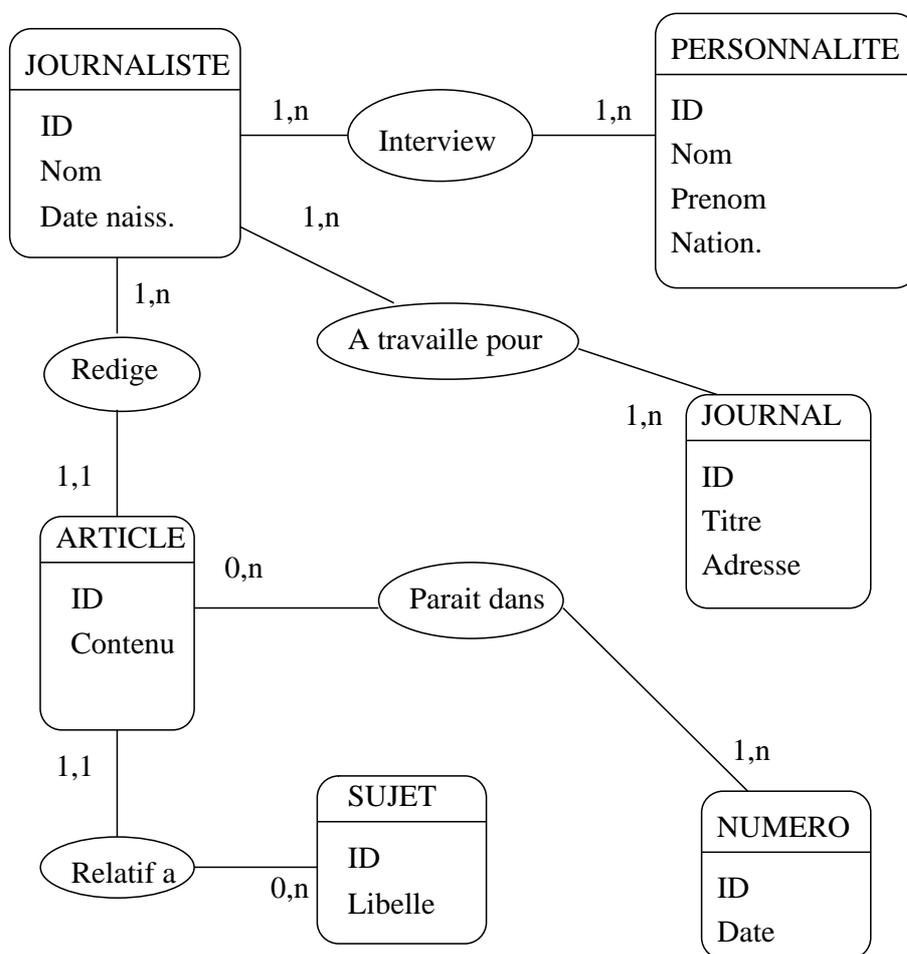


FIG. 1.3 – Journal

### 1.3 Rétro-conception

On trouve dans un SGBD relationnel les relations ci-dessous. Les clés primaires sont soulignées, mais pas les clés étrangères.

**IMMEUBLE** (Adresse, Nb-étages, Date-construction, Nom-Gérant)

**APPART** (Adresse, Numéro, Type, Superficie, Etage)

**PERSONNE** (Nom, Age, Code-Profession)

**OCCUPANT** (Adresse, Numéro-Appart, Nom-Occupant, Date-arrivée, Date-départ)

**PROPRIÉTÉ** (Adresse, Nom-Propriétaire, Quote-part)

**TYPE-APPART** (Code, Libellé)

**PROFESSION** (Code, Libellé)

**Exercice A** : Identifier les clés étrangères dans chaque relation.

**Exercice B** : Reconstruire le schéma E/A.

**Exercice C** : Existe-t-il des contraintes d'intégrité ? Lesquelles ?

**Exercice D** : Certaines données du schéma relationnel résultent-elles d'optimisation ?

---

## Chapitre 2

# Algèbre Relationnelle

### 2.1 Sélection et Projection

Soit la relation

| PERSONNE  |     |             |
|-----------|-----|-------------|
| Nom       | Age | Ville       |
| Marc      | 29  | Paris       |
| Catherine | 32  | Lyon        |
| Sophie    | 54  | Paris       |
| Claude    | 13  | Montpellier |
| Serge     | 40  | Lyon        |

**Exercice A :** Donnez les résultats des requêtes suivantes :

**Requête 1 :**  $\sigma_{Age=30}(PERSONNE)$  (sélection)

**Requête 2 :**  $\pi_{Age}(PERSONNE)$  (projection)

**Requête 3 :**  $\pi_{Age}(\sigma_{Nom \neq 'Serge'}(PERSONNE))$  (projection, sélection)

**Exercice B :** Exprimez les requêtes suivantes en algèbre relationnelle :

**Requête 1 :** les personnes (nom, âge, ville) qui habitent Paris.

**Requête 2 :** les personnes (nom, âge, ville) qui ont moins de 30 ans.

**Requête 3 :** les villes dans la relation PERSONNE.

**Requête 4 :** les noms des personnes habitant à Paris.

### 2.2 Jointure relationnelle

**Exercice A :** Soient **R** et **S** les relations

| R |   | S |   |
|---|---|---|---|
| A | B | B | C |
| a | b | b | c |
| a | f | e | a |
| c | b | b | d |
| d | e | g | b |

où les attributs A, B, C sont définis sur les domaines des lettres de l'alphabet.

Donnez le résultat des requêtes suivantes :

**Requête 1 :**  $R \bowtie S$  (jointure naturelle).

**Requête 2 :**  $\sigma_{A=C}(\rho_{B/B'}(R) \times S)$  (équi-jointure).

**Requête 3 :**  $R \bowtie\ltimes S = \pi_R(R \bowtie S)$  (semijoin).

**Exercice B :** Est-ce que les équations suivantes sont vraies ?

$$\pi_{A,B}(R \bowtie S) = R \tag{2.1}$$

$$\pi_{B,C}(R \bowtie S) = S \tag{2.2}$$

## 2.3 Auto-Jointure et Renommage

Soit  $T(A,B)$  une relation où A et B prennent leurs valeurs dans le même domaine. Supposons qu'on veuille construire une relation  $TS$  contenant les seuls n-uplets  $\langle a, b \rangle$  tels que  $\langle b, a \rangle$  est également un n-uplet de  $T$ . Par exemple :

| T |   | TS |   |
|---|---|----|---|
| A | B | A  | B |
| a | b | a  | b |
| a | c | b  | a |
| c | a | a  | c |
| c | d | c  | a |
| b | a |    |   |
| e | f |    |   |

Exprimez cette opération par une expression de l'algèbre relationnelle.

---

## Chapitre 3

# Algèbre - SQL : Employés - Départements

### 3.1 Schéma

Les exemples suivants sont tirés des sources de la société Oracle.

#### 3.1.1 Relation des Employés (EMP)

**EMP**(ENO, ENOM, PROF, DATEEMB, SAL, COMM, DNO)

**ENO** : numéro d'employé, clé

**ENOM** : nom de l'employé

**PROF** : profession (directeur n'est pas une profession)

**DATEEMB** : date d'embauche

**SAL** : salaire

**COMM** : commission (un employé peut ne pas avoir de commission)

**DNO** : numéro de département auquel appartient l'employé

#### 3.1.2 Relation des Départements (DEPT)

**DEPT**(DNO, DNOM, DIR, VILLE)

**DNO** : numéro de département, clé

**DNOM** : nom du département

**DIR** : directeur du département

**VILLE** : lieu du département (ville)

### 3.2 Opérations Algébriques

Soit l'exemple suivant :

|     | ENO | ENOM | PROF       | DATEEMB | SAL  | COMM | DNO |
|-----|-----|------|------------|---------|------|------|-----|
| EMP | 10  | Joe  | Ingénieur  | 1.10.93 | 4000 | 3000 | 3   |
|     | 20  | Jack | Technicien | 1.5.88  | 3000 | 2000 | 2   |
|     | 30  | Jim  | Vendeur    | 1.3.80  | 5000 | 5000 | 1   |
|     | 40  | Lucy | Ingénieur  | 1.3.80  | 5000 | 5000 | 3   |

|      | DNO | DNOM          | DIR | VILLE    |
|------|-----|---------------|-----|----------|
| DEPT | 1   | Commercial    | 30  | New York |
|      | 2   | Production    | 20  | Houston  |
|      | 3   | Développement | 40  | Boston   |

**Exercice A :** Calculer  $\sigma_{sal < 5000}(EMP)$ .

**Exercice B :** Calculer  $EMPbis = \rho_{ENO/ENO'}(\pi_{ENO,COMM}(EMP))$

**Exercice C :** Calculer  $\pi_{ENO,SAL}(EMP) \bowtie_{SAL=COMM} (EMPbis)$

**Exercice D :** Exprimer par une phrase ce qu'on obtient en évaluant les requêtes précédentes.

**Exercice E :** Quelle est l'expression de l'algèbre relationnelle qui permettrait d'obtenir le nom et la profession de l'employé de numéro 10.

**Exercice F :** Idem pour la liste des noms des employés qui travaillent à New York.

**Exercice G :** Idem pour avoir le nom du directeur du département "Commercial".

### 3.3 Requêtes

- Exprimer les requêtes Q1 à Q18 à l'aide de l'algèbre relationnelle.
- Exprimer en SQL les requêtes Q1 à Q24.

#### 3.3.1 Interrogation d'une seule Relation

**Requête 1 :** Donner tous les n-uplets de DEPT.

**Requête 2 :** Donner tous les n-uplets de EMP.

**Requête 3 :** Donner les noms et les salaires des employés.

**Requête 4 :** Donner les professions des employés (après élimination des duplicats).

**Requête 5 :** Donner les dates d'embauche des techniciens.

#### 3.3.2 Jointures

**Requête 6 :** Faire le produit cartésien entre EMP et DEPT.

**Requête 7 :** Donner les noms des employés et les noms de leur département.

- Requête 8 :** Donner les numéros des employés travaillant à BOSTON.
- Requête 9 :** Donner les noms des directeurs des départements 1 et 3. Attention : directeur n'est pas une profession !
- Requête 10 :** Donner les noms des employés travaillant dans un département avec au moins un ingénieur.
- Requête 11 :** Donner le salaire et le nom des employés gagnant plus qu'un (au moins un) ingénieur.
- Requête 12 :** Donner le salaire et le nom des employés gagnant plus que **tous les ingénieurs**.
- Requête 13 :** Donner les noms des employés et les noms de leurs directeurs.
- Requête 14 :** Trouver les noms des employés ayant le même directeur que JIM. Attention : un employé peut être directeur de plusieurs départements.
- Requête 15 :** Donner le nom et la date d'embauche des employés embauchés avant leur directeur ; donner également le nom et la date d'embauche de leur directeur.
- Requête 16 :** Donner les départements qui n'ont pas d'employés.
- Requête 17 :** Donner les noms des employés du département COMMERCIAL embauchés le même jour qu'un employé du département PRODUCTION.
- Requête 18 :** Donner les noms des employés embauchés avant *tous* les employés du département 1.
- Requête 19 :** Donner les noms des employés ayant le même emploi et le même directeur que JOE.

### 3.3.3 Valeurs Nulles, Tris, Groupes, Agrégats et Expressions

- Requête 20 :** Donner la liste des employés ayant une commission.
- Requête 21 :** Donner les noms, emplois et salaires des employés par emploi croissant et, pour chaque emploi, par salaire décroissant.
- Requête 22 :** Donner le salaire moyen des employés.
- Requête 23 :** Donner le nombre d'employés du département PRODUCTION.
- Requête 24 :** Les numéros de département et leur salaire maximum ?
- Requête 25 :** Donner les noms des employés ayant le salaire maximum de chaque département.
- Requête 26 :** Les professions et leur salaire moyen ?
- Requête 27 :** Le salaire moyen le plus bas (par profession) ?
- Requête 28 :** Donner les emplois ayant le salaire moyen le plus bas ; donnez aussi leur salaire moyen.

---

## Chapitre 4

# Algèbre - SQL : Appartements - Écoles

### 4.1 Schéma

**IMMEUBLE** (ADI, NBETAGES, DATEC, PROP)  
**APPIM** (ADI, NAPR, OCCUP, TYPE, SUPER, ETAGE)  
**PERSONNE** (NOM, AGE, PROF, ADR, NAPR)  
**ÉCOLE** (NOMECC, ADEC, NBCLASSES, DIR)  
**CLASSE** (NOMECC, NCL, MAITRE, NBEL)  
**ENFANT** (NOMP, PRENOM, AN, NOMECC, NCL)

avec la signification suivante :

#### 1. Relation **IMMEUBLE**

**ADI** : adresse d'immeuble, clé ; on fait l'hypothèse pour simplifier, que l'adresse identifie de manière unique un immeuble

**NBETAGES** : nombre d'étages d'un immeuble

**DATEC** : date de construction

**PROP** : nom du propriétaire de l'immeuble qui est une personne

#### 2. Relation **APPIM** (Appartement)

**ADI** : adresse d'immeuble

**NAPR** : numéro d'appartement

**OCCUP** : occupant de l'appartement (nom de la personne)

**TYPE** : type de l'appartement (Studio, F2, ...)

**SUPER** : superficie de l'appartement

**ETAGE** : étage où se situe l'appartement

#### 3. Relation **PERSONNE**

**NOM** : nom de personne, clé ; on fait l'hypothèse pour simplifier, que ce nom est unique sur l'ensemble des personnes que l'on considère dans la base

**AGE** : âge de la personne

**PROF** : profession de la personne (directeur d'école n'est pas une profession)

**ADR** : adresse de la résidence d'une personne, il s'agit d'un immeuble

**NAPR** : numéro d'appartement

#### 4. Relation ÉCOLE

**NOME** : nom d'une école, clé

**ADEC** : adresse d'une école

**NBCLASSES** : nombre de classes

**DIR** : nom du directeur

#### 5. Relation CLASSE

**NOME** : nom d'une école

**NCL** : nom de la classe, e.g., CP1, CE2, CE3, etc...

**MAITRE** : nom de l'instituteur

**NBEL** : nombre d'élèves dans la classe

#### 6. Relation ENFANT

**NOMP** : nom de la personne responsable de l'enfant, clé e.g., père, mère etc...

**PRENOM** : prénom de l'enfant

**AN** : année de naissance

**NOME** : nom d'une école

**NCL** : nom de la classe

La relation **IMMEUBLE** décrit un ensemble d'immeubles. Chaque immeuble a un propriétaire. La relation **APPIM** décrit pour chaque immeuble l'ensemble des appartements qui le compose. Chaque appartement peut héberger plusieurs personnes mais il y en a une qui est responsable (par exemple le locataire) et qui est désignée par le constituant **OCCUP**. Si l'appartement est inoccupé, ce constituant prend la valeur **NULL**. La relation **PERSONNE** décrit un ensemble de personnes. **ADR** et **NAPR** représentent l'adresse où réside une personne. Une personne peut avoir plusieurs enfants décrits par la relation **ENFANT**. Pour simplifier, on ne considère que les enfants allant à l'école primaire. Les écoles et les classes sont décrites dans les relations **ÉCOLE** et **CLASSE**. On suppose ici que toutes ces relations vérifient les contraintes d'intégrité de base. Par exemple, une personne répertoriée dans la relation **CLASSE** en tant que maître exerce bien la profession d'instituteur dans la relation **PERSONNE**. De même, un directeur d'école est un instituteur.

## 4.2 Requêtes

Exprimer les requêtes suivantes à l'aide de l'algèbre relationnelle, puis les traduire en SQL.

**Requête 1** : Donner l'adresse des immeubles ayant plus de 10 étages et construits avant 1970.

**Requête 2** : Donner les noms des personnes qui habitent dans un immeuble dont ils sont propriétaires (occupants et habitants).

**Requête 3** : Donner les noms des personnes qui ne sont pas propriétaires.

**Requête 4** : Donner les adresses des immeubles possédés par des informaticiens dont l'âge est inférieur à 40 ans.

**Requête 5 :** Donner la liste des *occupants* (nom, âge, profession) des immeubles possédés par DUPONT.

**Requête 6 :** Donner le nom et la profession des propriétaires d'immeubles où il y a des appartements vides.

**Requête 7 :** Donner les noms des maîtres qui habitent dans le même immeuble (à la même adresse) qu'au moins un de leurs élèves (on suppose que les enfants vivent sous le même toit que leur responsable).

**Requête 8 :** Donner l'adresse de l'immeuble, la date de construction, le type d'appartement et l'étage où habitent chacun des maîtres des enfants de DUPONT.

**Requête 9 :** Donner le nom et l'âge des maîtres qui habitent dans un immeuble dont le propriétaire est responsable d'un de leurs élèves.

**Requête 10 :** Donner le nom et l'âge des personnes qui sont propriétaires mais qui ne sont ni maître ni directeur d'école (des classes et écoles répertoriées dans ECOLE et CLASSE).

### 4.3 Mise à jour

**Requête 11 :** Ajouter un enfant de nom **np**, de prénom **e**, né en **a** et l'inscrire à la classe **c** de l'école **ec**.

### 4.4 Contraintes

Indiquer de la façon la plus formelle possible certaines contraintes que les données de la base doivent respecter pour être conformes à la réalité modélisée ici.

---

## Chapitre 5

# SQL - Algèbre : Cinémas - Films

### 5.1 Schéma

Les exemples suivants sont tirés du livre *Foundations of Databases* de S. Abiteboul, R. Hull et V. Vianu.

SALLE (Nom, Horaire, Titre)

FILM (Titre, Réalisateur, Acteur)

PRODUIT (Producteur, Titre)

VU (Spectateur, Titre)

AIME (Spectateur, Titre)

Un film est réalisé par un metteur en scène mais peut être financé par plusieurs Producteurs. Un Spectateur peut aimer un film sans l'avoir vu.

### 5.2 Requêtes

Écrire les requêtes suivantes en algèbre relationnelle .

#### 5.2.1 Interrogation d'une seule relation

**Requête 1 :** Dans quelle salle et à quelle heure peut on voir le film "Mad Max" ?

**Requête 2 :** Quels sont les films réalisés par Orson Welles ?

**Requête 3 :** Quels sont les Acteurs du film "Ran" ?

#### 5.2.2 Jointures

**Requête 4 :** Dans quelles salles peut-on voir un film avec Simone Signoret ?

**Requête 5 :** Dans quelles salles peut on voir Marlon Brando après 16h ?

**Requête 6 :** Quels sont les Acteurs qui ont produit un film ?

**Requête 7 :** Quels sont les Acteurs qui ont produit un film dans lequel ils jouent ?

**Requête 8 :** Quels sont les Acteurs qui ont produit et réalisé un même film ?

**Requête 9 :** Quels sont les Producteurs qui regardent les films qu'ils ont produits ?

### 5.2.3 Difference

**Requête 10 :** Quels films ne passent en ce moment dans aucune salle ?

**Requête 11 :** Quels Spectateurs aiment un film qu'ils n'ont pas vu ?

**Requête 12 :** Qui n'aime aucun film qu'il a vu ?

**Requête 13 :** Qui n'a produit aucun film de Doillon ?

**Requête 14 :** Qui a produit un film qui ne passe dans aucune salle ?

### 5.2.4 Division

**Requête 15 :** Quels Spectateurs ont vu tous les films ? (ou Spectateurs pour lesquels il n'existe pas un film qu'ils n'ont pas vu)

**Requête 16 :** Quels Acteurs jouent dans tous les films de Welles ? (ou Acteurs pour lesquels il n'existe pas un film de Welles qu'ils n'ont pas joué)

**Requête 17 :** Quels sont les Spectateurs qui aiment tous les films qu'ils ont vu ? (ou Spectateurs pour lesquels il n'existe pas un film qu'ils ont vu et qu'ils n'ont pas aimé)

**Requête 18 :** Quels sont les Producteurs qui voient tous les films qu'ils ont produit ? (ou Producteurs pour lesquels il n'existe pas un film qu'ils ont produit et qu'ils n'ont pas vu)

**Requête 19 :** Quels Producteurs voient tous les films de Kurosawa ? (ou Producteurs pour lesquels il n'existe pas un film de Kurosawa qu'ils n'ont pas vu)

---

## Chapitre 6

# Organisation Physique

**Exercice A :** On prend ici comme exemple la relation **Directeur** (*nom\_directeur, nom\_film*).

1. Organisation Séquentielle : Expliquer l'organisation séquentielle et représenter un exemple sur la relation Directeur. Montrer le fichier après une insertion et après quelques suppressions d'articles.
2. Organisation Indexée : Montrer des exemples d'index non dense (primaire) et dense (secondaire) sur la relation Directeur.

**Exercice B :** Construire un index sur la date de naissance des musicien (arbre B, ordre 2) :

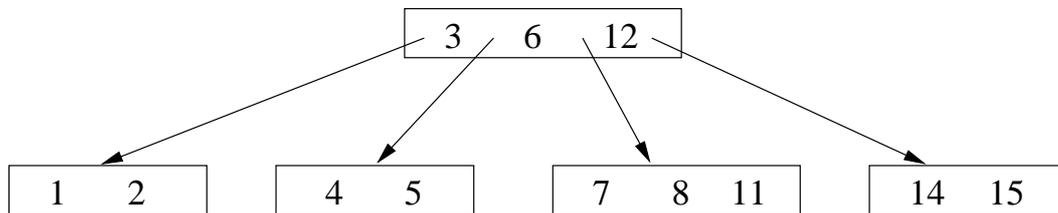
|            |      |
|------------|------|
| Monteverdi | 1589 |
| Couperin   | 1668 |
| Bach       | 1685 |
| Rameau     | 1684 |
| Debussy    | 1862 |
| Ravel      | 1875 |
| Mozart     | 1756 |
| Faure      | 1856 |

**Exercice C :** Construire un index sur les noms des musicien (arbre B, ordre 2).

**Exercice D :** Construire un arbre B+ d'ordre 2 sur les numéros de département.

|    |         |
|----|---------|
| 3  | Allier  |
| 36 | Indre   |
| 18 | Cher    |
| 9  | Ariège  |
| 11 | Aude    |
| 12 | Aveyron |
| 73 | Savoie  |
| 55 | Meuse   |
| 46 | Lot     |
| 39 | Jura    |
| 81 | Tarn    |
| 25 | Doubs   |
| 15 | Cantal  |
| 51 | Marne   |
| 42 | Loire   |

**Exercice E :** Soit le fichier séquentiel suivant (on ne donne pour chaque article du fichier que la clé sur laquelle on construit l'arbre) : 1 15 3 12 6 4 11 7 2 5 14 8 9 17 10 13 16. L'index en arbre B d'ordre 2 après l'insertion des clés 1 15 3 12 6 4 11 7 2 5 14 8 est montré dans la figure suivante :



1. Donnez l'arbre résultant après l'insertion de tous les articles du fichier séquentiel.
2. Combien de nœuds différents (racine et feuilles compris) doit-on parcourir dans l'index pour répondre à la requête qui cherche les articles dont la clé appartient à l'intervalle [5,10].

**Exercice F :** Soit les fichiers séquentiels suivants (on ne donne pour chaque article du fichier que la clé sur laquelle on construit l'arbre) :

- 5, 29, 17, 68, 60, 43, 10, 11, 12, 20, 55, 30, 40, 50, 25.
- 100, 29, 170, 70, 600, 430, 99, 11, 13, 21, 550, 30, 400, 50, 25
- 2 15 30 28 12 4 18 19 24 29 13 27 9 20 3 32 21 23

1. Construire un index en arbre B d'ordre 2 pour chacun des fichier.
2. Construire un index en arbre B+ d'ordre 2 pour chacun des fichier.

---

## Chapitre 7

# Algorithmes de Jointure

Soit les relations suivantes :

- **Directeur**(*nom\_directeur*, *nom\_film*)
- **Acteur**(*nom\_film*, *nom\_acteur*)

Soit la requête suivante :

```
SELECT nom_directeur, nom_acteur
FROM Directeur Join Acteur
WHERE Directeur.nom_film = Acteur.nom_film
```

Pour évaluer cette requête, il faut calculer la jointure **Directeur**  $\bowtie$  **Acteur**. Pour l'exécution des jointures, décrivez et évaluez la complexité de l'algorithme avec boucles imbriquées et avec tri-fusion. Dans les deux cas on tiendra compte des mouvements entre mémoire centrale et disque et on évaluera le nombre d'entrées-sorties.

**Exercice A :** *Algorithme avec boucles imbriquées ;*

**Exercice B :** *Algorithme avec tri-fusion ;*

---

## Chapitre 8

# Optimisation de Requêtes

**Exercice A : Soit la base STATION DE SKI contenant les relations suivantes :**

- hotel (noms, nomh, categorie, adresse, tel, nb\_chambres) (1000 pages, chaque page contient 10 tuples, 5 catégories différentes).
- station (noms, gare) (300 pages, chaque page contient 10 tuples).
- activite (type\_activite, noms) (100 pages, chaque page contient 10 tuples).

**On suppose de plus l'existence des index suivants (arbres B+) :**

- Index sur le couple (noms, nomh) de la relation hotel (26 pages).
- Index sur l'attribut categorie de hotel (33 pages).

**Hypothèses :**

- Chaque catégorie comprend un nombre égal d'hôtels.
- Chaque station comprend un nombre égal d'hôtels.
- Le nombre d'E/S lors du parcours de l'arborescence des index est *négligeable*.

**Requête 1 :** adresse, numéro de téléphone et nombre de chambres des hôtels de catégorie 3 dans la station de nom 'pesey'.

```
SELECT adresse, tel, nb_chambres
FROM hotel
WHERE noms='pesey' AND categorie=3;
```

1. Calculer le nombre d'E/S effectuées lors de l'évaluation de cette requête de sélection par un parcours séquentiel de la relation hotel.
2. Calculer le nombre d'E/S effectuées lors de l'évaluation de cette requête en utilisant l'index sur categorie.
3. Calculer le nombre d'E/S effectuées lors de l'évaluation de cette requête en utilisant l'index sur noms, nomh.
4. Conclure

**Requête 2 :** Soit le schéma suivant :

```
CREATE TABLE FILM (
    TITRE VARCHAR2(32),
    REALISATEUR VARCHAR2(32),
    ACTEUR VARCHAR2(32)
);
CREATE TABLE VU (
    SPECTATEUR VARCHAR2(32),
    TITRE VARCHAR2(32)
);
```

Soit la requête SQL :

```
SELECT ACTEUR, REALISATEUR
FROM FILM, VU
WHERE FILM.TITRE=VU.TITRE
```

Dans chacun des cas suivants, donner l'algorithme de jointure de ORACLE (par EXPLAIN, un arbre d'exécution commenté, une explication textuelle ou tout autre moyen de votre choix) :

1. Il n'existe pas d'index sur TITRE ni dans FILM ni dans VU,
2. Il existe un index sur TITRE dans FILM seulement.
3. Il existe un index sur TITRE dans les deux relations.

**Requête 3 :** Soit la requête :

```
SELECT e.enom, d.dnom
FROM emp e, dept d
WHERE e.dno = d.dno
AND e.sal = 10000
```

sur la relation **EMP** de schéma (*EMPNO, SAL, MGR, DNO*). Cette requête affiche le nom des employés dont le salaire (*SAL*) est égal à 10000, et celui de leur département. Indiquez le plan d'exécution dans chacune des hypothèses suivantes.

1. Index sur *DEPT(Dno)* et sur *EMP(Sal)*
2. Index sur *EMP(Sal)* seulement.
3. Index sur *EMP(Dno)* et sur *EMP(Sal)*
4. Voici une autre requête, légèrement différente. Plan d'exécution s'il n'y a pas d'index.

```
SELECT e.enom
FROM emp e, dept d
WHERE e.dno = d.dno
AND d.ville = 'Paris'
```

5. Que pensez-vous de la requête suivante par rapport à la précédente ?

```
SELECT e.enom
FROM emp e
WHERE e.dno IN (SELECT d.dno
                FROM Dept d
                WHERE d.Ville = 'Paris')
```

Voici le plan d'exécution donné par ORACLE :

```
0 SELECT STATEMENT
  1 MERGE JOIN
    2 SORT JOIN
      3 TABLE ACCESS FULL EMP
    4 SORT JOIN
      5 VIEW
        6 SORT UNIQUE
          7 TABLE ACCESS FULL DEPT
```

Qu'en dites vous ?

**Requête 4 :** Sur le même schéma, voici maintenant la requête suivante.

```
SELECT *
FROM EMP X WHERE X.SAL IN (SELECT SAL
                          FROM EMP
                          WHERE EMP.EMPNO=X.MGR)
```

Cette requête cherche les employés dont le salaire (*SAL*) est égal à celui de leur patron (*MGR*). On donne le plan d'exécution avec Oracle (outil EXPLAIN) pour cette requête dans deux cas : (i) pas d'index, (ii) un index sur le salaire et un index sur le numéro d'employé. Expliquez dans les deux cas ce plan d'exécution (éventuellement en vous aidant d'une représentation arborescente de ce plan d'exécution).

1. Pas d'index.

Plan d'exécution

-----

```
0 FILTER
  1 TABLE ACCESS FULL EMP
  2 TABLE ACCESS FULL EMP
```

2. Index empno et index sal.

Plan d'exécution

-----

```
0 FILTER
  1 TABLE ACCESS FULL EMP
  2 AND-EQUAL
    3 INDEX RANGE SCAN I-EMPNO
    4 INDEX RANGE SCAN I-SAL
```

3. Dans le cas où il y a les deux index (salaire et numéro d'employé) et où la requête est :

```
SELECT *
  FROM EMP X
 WHERE X.SAL = (SELECT SAL
                FROM EMP
                WHERE EMP.EMPNO=X.MGR)
```

on a le plan d'exécution suivant :

Plan d'exécution

-----

```
0 FILTER
  1 TABLE ACCESS FULL EMP
  2 TABLE ACCESS ROWID EMP
  3 INDEX RANGE SCAN I-EMPNO
```

Expliquez-le.

**Requête 5 :** On reprend le schéma CINEMA donné dans le cours, **mais on ne sais plus quels index existent.**

Questions :

1. Donner l'ordre SQL pour la requête : *Quels sont les films d'Hitchcock visibles après 20h00 ?*
2. Donner l'expression algébrique correspondante et proposez un arbre de requête qui vous paraît optimal.
3. Sous ORACLE, l'outil EXPLAIN donne le plan d'exécution suivant :

```
0 SELECT STATEMENT
  1 MERGE JOIN
    2 SORT JOIN
      3 NESTED LOOPS
        4 TABLE ACCESS FULL ARTISTE
```

```

5 TABLE ACCESS BY ROWID FILM
6 INDEX RANGE SCAN IDX-ARTISTE-ID
7 SORT JOIN
8 TABLE ACCESS FULL SEANCE

```

Commentez le plan donné par EXPLAIN. Pourrait-on améliorer les performances de cette requête ?

**Requête 6 :** Soit le schéma suivant :

```

CREATE TABLE Artiste (
    ID-artiste    NUMBER(4),
    Nom           VARCHAR2(32),
    Adresse      VARCHAR2(32)
);

CREATE TABLE Film (
    ID-film      NUMBER(4),
    Titre       VARCHAR2(32),
    Année       NUMBER(4),
    ID-réalisateur NUMBER(4)
);

CREATE TABLE Joue (
    ID-artiste    NUMBER(4),
    ID-film       NUMBER(4)
);

```

Questions :

1. Donner l'ordre SQL pour la requête : *Afficher le nom des acteurs et le titre des films où ils ont joué.*
2. Donner l'expression algébrique correspondante.
3. Quel est à votre avis le plan d'exécution dans s'il n'existe que deux index, un sur FILM(ID-réalisateur), et un sur ARTISTE(ID-artiste) ?
4. Idem, avec un index sur FILM(ID – Film), et un sur JOUE(ID – Artiste).
5. Idem, avec un index sur FILM(ID – Film), et un sur JOUE(ID – Film).

---

# Chapitre 9

## Concurrence

### 9.1 Sérialisabilité et recouvrabilité

#### 9.1.1 Graphe de sérialisation et équivalence des exécutions

Construisez les graphes de sérialisation pour les trois exécutions (histoires) suivantes. Indiquez les exécutions sérialisables et vérifiez si parmi les trois exécutions il y a des exécutions équivalentes.

1.  $H_1 : w_2[x] w_3[z] w_2[y] c_2 r_1[x] w_1[z] c_1 r_3[y] c_3$
2.  $H_2 : r_1[x] w_2[y] r_3[y] w_3[z] c_3 w_1[z] c_1 w_2[x] c_2$
3.  $H_3 : w_3[z] w_1[z] w_2[y] w_2[x] c_2 r_3[y] c_3 r_1[x] c_1$

#### 9.1.2 Recouvrabilité

Parmi les exécutions (histoires) suivantes, lesquelles sont recouvrables, lesquelles évitent les annulations en cascade et lesquelles sont strictes ? Indiquez s'il y a des exécutions sérialisables.

1.  $H_1 : r_1[x] w_2[y] r_1[y] w_1[x] c_1 r_2[x] w_2[x] c_2$
2.  $H_2 : r_1[x] w_1[y] r_2[y] c_1 w_2[x] c_2$
3.  $H_3 : r_1[y] w_2[x] r_2[y] w_1[x] c_2 r_1[x] c_1$

### 9.2 Contrôle de concurrence

#### 9.2.1 Verrouillage à 2 phases

Un ordonnanceur avec verrouillage à 2 phases reçoit la séquence d'opérations ci-dessous.

$H : r_1[x] r_2[y] w_3[x] w_1[y] w_1[x] w_2[y] c_2 r_3[y] r_1[y] c_1 w_3[y] c_3$

Indiquez l'ordre d'exécution établi par l'ordonnanceur, en considérant que les opérations bloquées en attente d'un verrou seront exécutées en priorité dès que le verrou devient disponible, dans l'ordre de leur blocage. On suppose que les verrous d'une transaction sont relâchés au moment du Commit.

### 9.2.2 Verrouillage, interblocage

On considère l'exécution concurrente suivante :

$H : r_1[x] r_2[y] r_3[x] w_3[x] w_1[y] r_2[x] c_1 c_2 c_3$

1. Trouvez les conflits dans H et montrez que H n'est pas sérialisable.
2. Trouvez l'exécution obtenue à partir de H par verrouillage à deux phases. On considère que les verrous d'une transaction sont relâchés au Commit et qu'à ce moment les opérations bloquées en attente de verrou sont exécutées en priorité.
3. Le raisonnement suivant mène à une conclusion paradoxale.  
 Considérons une histoire quelconque H non-sérialisable, traitée par verrouillage à deux phases. H contient donc un cycle dans son graphe de sérialisation. Chaque arc  $T_i \rightarrow T_j$  de ce cycle provient d'une paire d'opérations conflictuelles de type "a<sub>i</sub>[x] avant b<sub>j</sub>[x]". Quand a<sub>i</sub>[x] s'exécute, elle prend le verrou sur x, ce qui bloquera b<sub>j</sub>[x].  
 Donc T<sub>i</sub> bloque T<sub>j</sub>, mais comme les arcs forment un cycle, il y aura un blocage circulaire, donc un interblocage. Conclusion : toute histoire non-sérialisable traitée par verrouillage à deux phases produit un interblocage ! Cependant, l'exemple précédent montre que cette conclusion est fautive, car H n'est pas sérialisable et elle ne produit pas d'interblocage. Expliquez pourquoi il n'y a pas d'interblocage et pourquoi le raisonnement ci-dessus est faux.
4. Montrez que H n'évite pas les annulations en cascade. Montrez qu'en avançant des Commit, H peut éviter les annulations en cascade.

### 9.2.3 Verrouillage hiérarchique

Considérons une relation **Compte**(Numéro, Titulaire, Solde) et trois opérations qui s'exécutent sur cette relation à deux niveaux de granularité : relation et enregistrement. Supposons que dans cette relation il y a deux comptes pour le titulaire "Dupont". Les trois opérations sont les suivantes :

- un programme qui lit tous les comptes ;
- un programme qui réalise un transfert entre les deux comptes de Dupont ;
- un programme qui lit tous les comptes, afin de trouver les comptes de Dupont et de créditer leur solde avec les intérêts de l'année.

1. Montrer que l'exécution suivante est une exécution concurrente des trois opérations ci-dessus. Identifier les données qui correspondent à chaque variable.  
 $H : r_1[z] r_2[z] w_1[x] r_3[x] w_3[x] c_2 w_1[y] r_3[y] c_1 w_3[y] c_3$
2. Trouver l'exécution obtenue par verrouillage hiérarchique à partir de H. On considère que le verrouillage hiérarchique utilise des verrous de type SIX (lecture et intention d'écriture), dans le cas présent pour le rajout des intérêts sur les comptes de Dupont.  
 Les verrous sont relâchés au Commit et les opérations bloquées en attente de verrou sont traitées en priorité, dans l'ordre de leur blocage.
3. Que se passe-t-il dans le verrouillage hiérarchique de H si on n'utilise pas de verrou SIX au niveau de la relation (mais seulement du S et du IX séparément) ?

## 9.3 Reprise après panne

### 9.3.1 Journalisation

Soit le journal physique ci-dessous, dans lequel on a marqué les opérations Commit et Abort réalisées :

$[T_1, x, 3], [T_2, y, 1], [T_1, z, 2], c_1, [T_2, z, 4], [T_3, x, 2], a_2, [T_4, y, 3], c_3, [T_5, x, 5]$

1. Indiquez le contenu de *liste\_commit*, *liste\_abort*, *liste\_active*.
2. En supposant qu'une nouvelle écriture vient de s'ajouter au journal, lesquelles des écritures suivantes sont compatibles avec une exécution stricte :  $[T_5, y, 4], [T_4, z, 3]$  ou  $[T_6, x, 1]$  ?
3. Quelles sont les entrées récupérables par l'algorithme de ramasse-miettes ?
4. Si les valeurs initiales des enregistrements étaient  $x = 1, y = 2, z = 3$ , et si une panne survenait à ce moment, quelles seraient les valeurs restaurées pour  $x, y$  et  $z$  après la reprise ?
5. En pratique, on stocke souvent dans chaque entrée aussi l'image avant de l'écriture (la valeur de l'enregistrement avant l'écriture). Aussi, les positions des Commit et Abort ne sont pas connues, mais seulement les listes de transactions *liste\_commit*, *liste\_abort* et *liste\_active* au moment courant (la fin du journal). Quel est dans ce cas le contenu du journal ? Les valeurs initiales sont celles précisées ci-dessus ( $x = 1, y = 2, z = 3$ ), ainsi que les listes courantes de transactions.
6. Expliquez le fonctionnement de l'algorithme de reprise avec annulation et avec répétition sur ce dernier journal.

## 9.4 Concurrence : Gestion Bancaire

Les trois programmes suivants peuvent s'exécuter dans un système de gestion bancaire. *Débit* diminue le solde d'un compte  $c$  avec un montant donné  $m$ . Pour simplifier, tout débit est permis (on accepte des découverts). *Crédit* augmente le solde d'un compte  $c$  avec un montant donné  $m$ . *Transfert* transfère un montant  $m$  à partir d'un compte source  $s$  vers un compte destination  $d$ . L'exécution de chaque programme démarre par un **Start** et se termine par un **Commit** (non montrés ci-dessous).

|   |  |   |
|---|--|---|
| <pre>Débit (c:Compte;       m:Montant) begin   t := Read(c);   Write(c, t-m); end</pre> | <pre>Crédit (c:Compte;        m:Montant) begin   t = Read(c);   Write(c, t+m); end</pre> | <pre>Transfert (s,d:Compte;           m:Montant) begin   Débit(s,m);   Crédit(d,m); end</pre> |
|---|--|---|

Le système exécute en même temps les trois opérations suivantes :

- (1) un transfert de montant 100 du compte A vers le compte B
- (2) un crédit de 200 pour le compte A
- (3) un débit de 50 pour le compte B

1. Écrire les transactions  $T_1, T_2$  et  $T_3$  qui correspondent à ces opérations. Montrer que l'histoire  $\mathbf{H} : r_1[A] r_3[B] w_1[A] r_2[A] w_3[B] r_1[B] c_3 w_2[A] c_2 w_1[B] c_1$  est une exécution concurrente de  $T_1, T_2$  et  $T_3$ .
2. Mettre en évidence les conflits dans  $\mathbf{H}$  et construire le graphe de sérialisation de cette histoire.  $\mathbf{H}$  est-elle sérialisable ?  $\mathbf{H}$  est-elle recouvrable ?

3. Quelle est l'exécution  $H'$  obtenue à partir de  $H$  par verrouillage à deux phases ? On suppose que les verrous d'une transaction sont relâchés après le Commit de celle-ci. Une opération bloquée en attente d'un verrou bloque le reste de sa transaction. Au moment du relâchement des verrous, les opérations en attente sont exécutées en priorité, dans l'ordre de leur blocage.
- Si au début le compte A avait un solde de 100 et B de 50, quel sera le solde des deux comptes après la reprise si une panne intervient après l'exécution de  $w_1[B]$  ?

## 9.5 Concurrence : Agence environnementale

Une agence environnementale utilise une base de données relationnelle pour gérer des informations sur les véhicules. Le schéma est le suivant (les clés sont soulignées) :

**propriétaire**(nom, âge)

**véhicule**(immatriculation, kilométrage, nom, idmodèle) (nom = nom propriétaire)

**modèle**(idmodèle, crashtest, consommation)

**carburant**(désignation, cours, pollution)

**utilise**(idmodèle, désignation)

L'exécution suivante est reçue par le système de l'agence environnementale :

$H : r_1[x]r_2[y]r_3[x]w_2[y]w_1[x]r_3[y]r_1[z]w_2[z]w_1[y]c_1c_2w_3[z]c_3$

1. Parmi les programmes qui s'exécutent dans le système, il y a  $KilométrAge(i, k, a)$ , qui fixe pour le véhicule d'immatriculation  $i$  le kilométrage à  $k$  et l'âge du propriétaire à  $a$ . Montrez quelle transaction de  $H$  pourrait provenir de  $KilométrAge$ .  
On considère que les enregistrements (variables) de  $H$  sont des nuplets des relations de la base de données. On suppose que tout nuplet est accessible directement si l'on connaît sa clé.
2. Identifiez tous les conflits dans  $H$  et vérifiez si l'exécution est sérialisable en construisant le graphe de sérialisation.
3. A quel niveau de recouvrabilité se situe  $H$  (recouvrable, évitant les annulations en cascade, stricte) ?
4. Quelle est l'exécution obtenue par verrouillage à deux phases à partir de  $H$  ? Le relâchement des verrous d'une transaction se fait au Commit et à ce moment on exécute en priorité les opérations bloquées en attente de verrou, dans l'ordre de leur blocage.