

## Énoncé 3 : algèbre relationnelle

[guillaume.postic@univ-evry.fr](mailto:guillaume.postic@univ-evry.fr)

## Exercice 1

1. Oui

+-----+	
	NomC
+-----+	
	Jean
	Paul
	Vincent
+-----+	

Dans le modèle relationnel, il n'y a pas de doublon dans l'extension de la relation (l'extension est l'ensemble des uplets). Ici, Jean et Paul n'apparaissent donc qu'une seule fois. Le SQL ne respecte pas complètement le modèle relationnel : dans une requête SQL équivalente, Jean et Paul apparaîtraient deux fois chacun.

2. Oui

+-----+-----+		
	NomP	Couts
+-----+-----+		
	sable	300
	briques	1500
	parpaing	1150
	tuiles	1150
	parpaing	1200
	parpaing	1150
	ciment	125
	briques	1200
+-----+-----+		

Doublon : présent en SQL, pas dans le modèle relationnel

3. Non : union possible seulement si les deux relations ont les mêmes attributs. En SQL, le nombre d'attributs est la seule condition (leurs domaines de définition peuvent être différents).
4. Non : même raison que pour l'union.

Note : dans le SGBD MySQL, **INTERSECT** n'existe pas. Il faut passer par un **INNER JOIN** sur toutes les colonnes.

5. Oui. Le fournisseur Cima n'est pas présent dans la table Prix.

NomF
Abounayan
Preblocs
Samaco

6. Non (voir 3 et 4). Dans MySQL, **MINUS** n'existe pas : il faut utiliser **NOT IN**.

7. Oui

NomF
Cima

8. Oui : produit cartésien

NomF	AdresseF	NomF	NomP	Couts
Abounayan	92190 Meudon	Abounayan	sable	300
Cima	75010 Paris	Abounayan	sable	300
Preblocs	92230 Gennevilliers	Abounayan	sable	300
Samaco	75116 Paris	Abounayan	sable	300
Abounayan	92190 Meudon	Abounayan	briques	1500
Cima	75010 Paris	Abounayan	briques	1500
Preblocs	92230 Gennevilliers	Abounayan	briques	1500
Samaco	75116 Paris	Abounayan	briques	1500
Abounayan	92190 Meudon	Abounayan	parpaing	1150
Cima	75010 Paris	Abounayan	parpaing	1150
Preblocs	92230 Gennevilliers	Abounayan	parpaing	1150
Samaco	75116 Paris	Abounayan	parpaing	1150
Abounayan	92190 Meudon	Preblocs	tuiles	1150
Cima	75010 Paris	Preblocs	tuiles	1150
Preblocs	92230 Gennevilliers	Preblocs	tuiles	1150
Samaco	75116 Paris	Preblocs	tuiles	1150
Abounayan	92190 Meudon	Preblocs	parpaing	1200
Cima	75010 Paris	Preblocs	parpaing	1200
Preblocs	92230 Gennevilliers	Preblocs	parpaing	1200
Samaco	75116 Paris	Preblocs	parpaing	1200
Abounayan	92190 Meudon	Samaco	parpaing	1150
Cima	75010 Paris	Samaco	parpaing	1150
Preblocs	92230 Gennevilliers	Samaco	parpaing	1150

Samaco	75116 Paris	Samaco	parpaing	1150
Abounayan	92190 Meudon	Samaco	ciment	125
Cima	75010 Paris	Samaco	ciment	125
Preblocs	92230 Gennevilliers	Samaco	ciment	125
Samaco	75116 Paris	Samaco	ciment	125
Abounayan	92190 Meudon	Samaco	briques	1200
Cima	75010 Paris	Samaco	briques	1200
Preblocs	92230 Gennevilliers	Samaco	briques	1200
Samaco	75116 Paris	Samaco	briques	1200
+-----+-----+-----+-----+				

9. Oui

10. Non : aucune colonne pour faire la jointure (différence avec SQL)

11. Oui : jointure sur colonne NomF.

Note : c'est une jointure naturelle, donc la colonne NomF n'apparaît qu'une seule fois (à la différence de l'équi-jointure).

+-----+-----+-----+-----+				
NomF	AdresseF	NomP	CoutS	
+-----+-----+-----+-----+				
Abounayan	92190 Meudon	sable	300	
Abounayan	92190 Meudon	briques	1500	
Abounayan	92190 Meudon	parpaing	1150	
Preblocs	92230 Gennevilliers	tuiles	1150	
Preblocs	92230 Gennevilliers	parpaing	1200	
Samaco	75116 Paris	parpaing	1150	
Samaco	75116 Paris	ciment	125	
Samaco	75116 Paris	briques	1200	
+-----+-----+-----+-----+				

12. Oui : comme 11 mais seuls les attributs de Fournisseurs sont gardés.

+-----+-----+		
NomF	AdresseF	
+-----+-----+		
Abounayan	92190 Meudon	
Preblocs	92230 Gennevilliers	
Samaco	75116 Paris	
+-----+-----+		

13. Oui

NumCom	NomC	NomP	Qte
1	Jean	briques	5
2	Jean	ciment	10

14. Oui

NomC	AdresseC	Solde	NumCom	NomP	Qte
Jean	75006 Paris	-12000	1	briques	5
Jean	75006 Paris	-12000	2	ciment	10
Paul	75003 Paris	0	3	briques	3
Paul	75003 Paris	0	4	parpaing	9
Vincent	94200 Ivry	3000	5	parpaing	7

15. Oui

NomC	AdresseC	Solde
Jean	75006 Paris	-12000
Paul	75003 Paris	0
Vincent	94200 Ivry	3000

16. Oui

NomP
briques
ciment

17. Oui

NumCom	NomC	NomP	Qte	NomF	Couts
1	Jean	briques	5	Abounayan	1500
2	Jean	ciment	10	Samaco	125
1	Jean	briques	5	Samaco	1200

18. Oui

19. Oui

NomP	NomF	Couts
briques	Abounayan	1500
ciment	Samaco	125
briques	Samaco	1200

2. Formuler les requêtes suivantes en langage algébrique et en SQL :

Quels sont les noms des produits commandés par Jean ?

`/*1*/`

`SELECT NomP FROM Commandes WHERE NomC = 'Jean'`

$\pi_{\text{NomP}} (\sigma_{\text{NomC} = \text{'Jean'}} (\text{Commandes}))$

Quels sont les noms des fournisseurs qui fournissent les produits qui figurent dans les commandes de Paul ?

`/*2*/`

`SELECT p.NomF  
FROM Commandes c  
JOIN Prix p  
ON c.NomP = p.NomP  
WHERE c.NomC = 'Paul'`

`/* Avec NATURAL JOIN */`

`SELECT p.NomF  
FROM Commandes c  
NATURAL JOIN Prix p  
WHERE c.NomC = 'Paul'`

$\pi_{\text{NomF}} (\text{Prix} \bowtie \sigma_{\text{NomC} = \text{'Paul'}} (\text{Commandes}))$

Quelle est l'adresse des fournisseurs qui fournissent des parpaings à un coût strictement inférieur à 1200 ?

/\*3\*/

```
SELECT f.AdresseF
FROM Prix p
JOIN Fournisseurs f
ON p.NomF = f.NomF
WHERE p.NomP = 'parpaing'
AND p.Couts < 1200
```

$\pi_{\text{AdresseF}} ( \text{Fournisseurs} \bowtie \sigma_{\text{NomP} = \text{'parpaing'} \wedge \text{Couts} < 1200} (\text{Prix}) )$

Quels sont les noms et adresses des clients et des fournisseurs tels que le produit commandé lors d'une commande soit des briques ?

/\*4\*/

```
SELECT c.NomC, k.AdresseC, p.NomF, f.AdresseF
FROM Commandes c
JOIN Clients k
ON c.NomC = k.NomC
JOIN Prix p
ON c.NomP = p.NomP
JOIN Fournisseurs f
ON p.NomF = f.NomF
WHERE c.NomP = 'briques'
```

```
/* implicit inner join */
SELECT c.NomC, k.AdresseC, p.NomF, f.AdresseF
FROM Commandes c, Clients k, Prix p, Fournisseurs f
WHERE c.NomP = 'briques'
AND c.NomC = k.NomC
AND c.NomP = p.NomP
AND p.NomF = f.NomF
```

```
/* Avec NATURAL JOIN */
SELECT c.NomC, k.AdresseC, p.NomF, f.AdresseF
FROM Commandes c
NATURAL JOIN Clients k
NATURAL JOIN Prix p
NATURAL JOIN Fournisseurs f
WHERE c.NomP = 'briques'
```

$$\pi_{\text{AdresseF, NomC, AdresseC, NomF}} ( \text{Fournisseurs} \bowtie \sigma_{\text{NomP} = \text{'briques'}} ( \text{Prix} \bowtie (\text{Commandes} \bowtie \text{Clients}) ) )$$

## Exercice 2

1. Formuler les requêtes suivantes en langage algébrique et en SQL :

(a) Intervenants de la pièce "L'avare" ?

```
SELECT i.Intervenant
FROM Intervenants i
WHERE i.Titre = 'L avare';
```

$$\pi_{\text{Intervenant}} ( \sigma_{\text{Titre} = \text{'L avare'}} (\text{Intervenants}) )$$

(b) Intervenants qui n'interviennent pas dans la pièce "L'avare" ?

```
SELECT i.Intervenant
FROM Intervenants i
WHERE i.Intervenant NOT IN
    (SELECT Intervenant
     FROM Intervenants
     WHERE Titre = 'L avare');
```

$$\pi_{\text{Intervenant}} (\text{Intervenants} - ( \sigma_{\text{Titre} = \text{'L avare'}} (\text{Intervenants}) ) )$$

(c) Intervenants qui sont présents au moins dans un spectacle chaque semaine pendant la saison ?

```
SELECT i.Intervenant,
       COUNT(DISTINCT s.Semaine) AS compte,
       GROUP_CONCAT(DISTINCT s.Semaine) /* facultatif */
FROM Intervenants i
JOIN Spectacles s
ON i.Titre = s.Titre
GROUP BY i.Intervenant
HAVING compte = 4
```

$\pi_{\text{Intervenant}} (\sigma_{\text{Semaine} = 4} ( \text{Intervenant} \text{ } g_{\text{count}(\text{Semaine})} (\text{Intervenants} \bowtie \text{Spectacles}) ))$

$g$  : opérateur d'agrégation

L'algèbre relationnelle est basée sur la théorie des ensembles. Par définition, il n'y a pas de doublon dans un ensemble : tous les éléments de l'ensemble sont uniques. Le langage SQL ne respecte pas complètement le modèle relationnel, puisqu'il autorise plusieurs *tuples* identiques (sauf lorsqu'une contrainte d'unicité est explicitement appliquée dans le schéma, avec **UNIQUE**). D'où l'existence de la commande **DISTINCT** pour les requêtes.

Solution alternative sans **DISTINCT**, mais avec **GROUP BY** sur deux colonnes :

```
SELECT table1.Intervenant, COUNT(table1.Intervenant) AS compte
FROM (
  SELECT i.Intervenant, s.Semaine, COUNT(s.Semaine)
  FROM Intervenants i
  JOIN Spectacles s
  ON i.Titre = s.Titre
  GROUP BY i.Intervenant, s.Semaine
) table1
GROUP BY Intervenant
HAVING compte = 4
```



(d) Noms des salles libres au moins une semaine dans la saison ?

```
SELECT p.Salle, COUNT(DISTINCT p.Semaine)
FROM Places p
GROUP BY p.Salle
HAVING COUNT(DISTINCT p.Semaine) < 4;
```

$\pi_{\text{Salle}} (\sigma_{\text{Semaine} < 4} ( \text{Salle} \bowtie_{\text{count}(\text{Semaine})} \text{Places} ))$

IMPORTANT : les opérations d'agrégation ne font pas partie de l'algèbre relationnelle telle qu'elle a été initialement définie par Edgar F. Codd : elles ont été ajoutées dans une version étendue (voir *Database system concepts* Vol. 5, par Silberschatz, Korth et Sudarshan, 2002).

(e) A quelle date (semaine et jour) reste-t-il des places pour aller voir l'intervenant Dupont (et accessoirement dans quel titre) ?

```
SELECT i.Intervenant, s.Titre, s.Semaine, p.Jour, p.Disponibilite
FROM Places p
JOIN Spectacles s
ON p.Semaine = s.Semaine
AND p.Salle = s.Salle
JOIN Intervenants i
ON i.Titre = s.Titre
WHERE i.Intervenant = 'Paul' /* Choisir ici l'intervenant */
AND p.Disponibilite > 0
```

```
/* implicit inner join */
SELECT i.Intervenant, s.Titre, s.Semaine, p.Jour,
p.Disponibilite
FROM Places p, Spectacles s, Intervenants i
WHERE p.Semaine = s.Semaine
AND p.Salle = s.Salle
AND i.Titre = s.Titre
AND i.Intervenant = 'Paul' /* Choisir ici l'intervenant */
AND p.Disponibilite > 0
```

$\pi_{\text{Interv, Titre, Semaine, Jour, Dispo}} ( ( \text{Spectacles} \bowtie \sigma_{\text{Intervenant} = \text{'Paul'}} ( \text{Intervenants} ) ) \bowtie \sigma_{\text{Dispo} > 0} ( \text{Places} ) )$

(f) Intervenants qui interviennent dans toutes les salles au cours de la saison ?

```
SELECT i.Intervenant, COUNT(DISTINCT s.Salle)
FROM Intervenants i
JOIN Spectacles s
ON i.Titre = s.Titre
GROUP BY i.Intervenant
HAVING COUNT(DISTINCT s.Salle) = 3
```

$$\pi_{\text{Intervenant}} (\sigma_{\text{Salle} = 3} ( \text{Intervenant} \bowtie_{\text{count}(\text{Salle})} (\text{Intervenants} \bowtie \text{Spectacles}) ))$$

2. Les représentations (séances) pour lesquelles personne n'a fait de réservation.