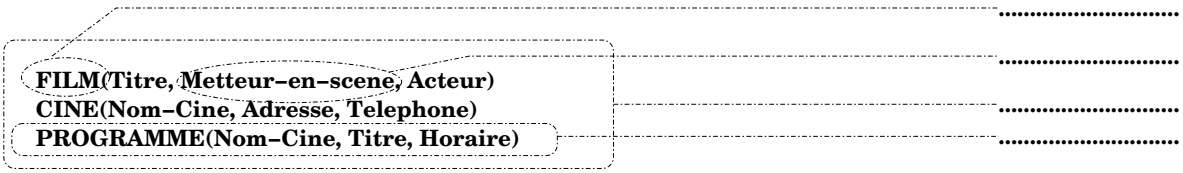
Feuilles de TD du cours de BD

Peip2.
Certains ex. ont été inspirés par le cours de N. Bidoit (L2), d'autres par des manuels.

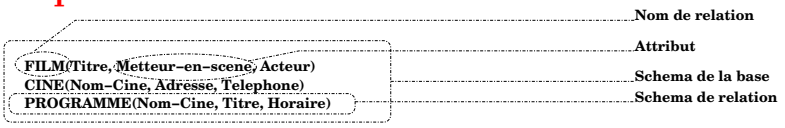
Exercice 0 : Questions de cours : modèle relationnel(20 mins)

Remplir la figure suivante :



FILM	Titre	Metteur–en–scene	Acteur
	Speed2	Jan de Bont	S. Bullock
	Speed2	Jan de Bont	J. Patrick
	Speed2	Jan de Bont	W. Dafoe
	Marion	M. Poirier	C. Tetard
	Marion	M. Poirier	M–F. Pisier

Réponse :



FILM	Titre	Metteur–en–scene	Acteur
	Speed2	Jan de Bont	S. Bullock
	Speed2	Jan de Bont	J. Patrick
	Speed2	Jan de Bont	W. Dafoe
	Marion	M. Poirier	C. Tetard
	Marion	M. Poirier	M–F. Pisier

Instance de relation

n–uplet ou tuple

Exercice 1 : modèle relationnel (10 mins)

1. Considérons le schéma de relation `STOCK(Habit, TailleNum)`, où
   Dom(`Habit`)=$\{Jupe, Jean\}$
   Dom(`TailleNum`)=$\{38, 40, 42\}$.

   (a) Combien de n-uplets sont possibles pour la relation `STOCK` ?
   Réponse : $|Dom(Habit)| \times |Dom(TailleNum)| = 2 \times 3 = 6$

   (b) Combien d'instances de cardinalité 0 de la relation `STOCK` existe-t-il ?
   Réponse : Une seule : la vide. Ce nombre est donné par $C_0^6 = \binom{6}{0} = 1$.

   (c) Combien d'instances de cardinalité 2 de la relation `STOCK` existe-t-il (sous la sémantique ensembliste du modèle relationnel) ?

Réponse : Il y en a $C_2^6 = \binom{6}{2} = 15$.

(d) Donnez une instance de la relation `STOCK` de cardinalité 2.

Réponse : Exemple :

STOCK	Habit	TailleNum
	Jupe	38
	Jupe	42

Exercice 2 : Schéma relationnel, clés, domaines... (20 mins)

Exercice 11 a,b,c,e du chapitre 4 dans le Manuel NSI de Terminale chez Hachette : `https://mesmanuels.fr/acces-libre/9782017189992`

Exercice 3 : Algèbre relationnelle : comprendre (30 mins)

Considérons l'instance suivante de bases de données :

R	A	B
	1	2
	2	2
	6	7
	4	7

S	A	B

T	A	B
	2	3
	4	7
	4	4

U	A	C
	1	2
	2	2

Donnez le résultat de chacune des requêtes suivantes :

1. $[R] \cup [S]$
   Réponse : même instance que R car S est vide.
2. $[R] \cup [U]$
   Réponse : une erreur car la requête est invalide.
3. $\Pi_C[U]$
   Réponse :

$\Pi_C[U]$	C
	2

4. $\sigma_{A\geq 2}[\pi_{A,C}[R \bowtie \rho_{B\to C}[T]]]$
   Réponse : Décomposons : $R \bowtie \rho_{B\to C}[T]]$ donne

$R \bowtie \rho_{B\to C}[T]]$	A	B	C
	2	2	3
	4	7	7
	4	7	4

Donc l'expression entière renvoie :

$\sigma_{A\geq 2}[\pi_{A,C}[R \bowtie \rho_{B\to C}[T]]]$	A	C
	2	3
	4	7
	4	4

5. $[[T] \cup [R]] - [[[T] - [R]] \cup [[R] - [T]]]$.
   Réponse :

$= [T] \cap [R] =$	A	B
	4	7

Exercice 4 : Arbres syntaxique pour l'algèbre (10 mins)

A partir d'une expression algébrique on peut construire un arbre syntaxique représentant la structure de cette formule (et une manière de l'évaluer "bottom-up" en partant des feuilles). A chaque expression correspond un arbre, et on peut choisir pour calculer une expression algébrique de la réécrire sous une forme plus efficace appliquant les règles de l'algèbre traditionnelle (ex : associativité, commutativité). L'arbre de gauche en Figure 1 permet ainsi de calculer l'expression

$((x * z) - (y * z)) + 2$ [1]. L'arbre de droite permet d'évaluer une expression équivalente.
Soient $R(a, b, c)$ et $S(c, d, e)$. Donner un arbre syntaxique (plan de requête) pour l'expression E : $\pi_{a,d}(\sigma_{a=b}(R \bowtie S))$. Donner aussi l'arbre syntaxique d'une expression équivalente à E.

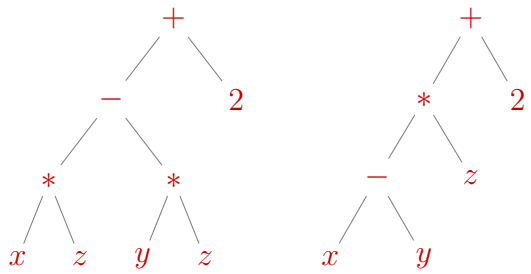
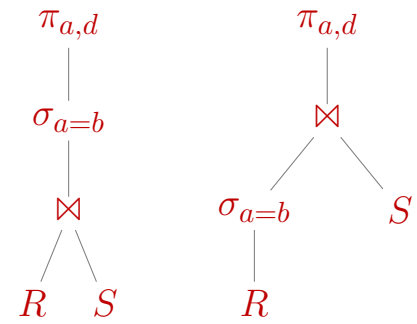


FIGURE 1 – Arbre syntaxique pour $(x * z - y * z) + 2$

Réponse :



Exercice 5 : Algèbre rel. : exprimer une requête (10 mins)

On considère le schéma de base de données suivant :
Produit (id_produit , nom_produit , id_fournisseur , id_categorie , prix_unitaire)
Fournisseur (id_fournisseur , compagnie , pays)
Categorie (id_categorie,nom_categorie)
dans lequel les champs id_fournisseur et id_categorie de Produit sont des clés étrangères vers les tables de même nom. On n'a pas précisé les types mais ils sont à peu près évidents...On supposera que les prix sont toujours indiqués en euros (modéliser intelligemment une base de données de multinationale prenant en compte des spécificités locales n'est pas du ressort de ce cours).
Une instance correspondante est donnée ci-dessous :

1. Remarque anecdotique, pour aller plus loin : noter que nous sommes habitués à la notation infixe, où l'opérateur binaire est "au milieu" . En notation préfixe, on place l'opérateur avant ses arguments, donc l'on écrirait cette expression + - * x z * y z 2. Un des avantages des notations préfixes et postfixes (appelées aussi polonaise et polonaise inverse) est qu'il n'y a plus besoin de parenthèses lorsque l'arité des opérateurs est connue. Lorsque l'on compose des fonctions, on utilise la notation préfixe (ex : f(g(2,h(4,5,6))). Certaines calculettes adoptent la notation polonaise inverse (postfixe).

A partir de l'arbre syntaxique on retrouve facilement toutes ces façons de noter l'expression : on va parcourir l'arbre en partant de la racine et en explorant les fils gauches en profondeur avant de passer au fils droit. La notation préfixe affiche un nœud lors de la première visite, donc avant son sous-arbre gauche. La notation infixe affiche un nœud entre ses deux sous-arbres (immédiatement si c'est une feuille). La notation postfixe affiche un nœud après son sous-arbre droit. La notion d'arbre de syntaxe est particulièrement utilisée dans les compilateurs, qu'il s'agisse de compiler (=transformer en code de bas niveau pour la machine) un programme java, C, python, ou une requête de base de données. On les retrouve aussi en traitement du langage naturel pour analyser les éléments grammaticaux d'une phrase.

Produit

id_produit	nom_produit	id_fournisseur	id_categorie	prix_unitaire
1	Calissons	1	8	17.99
2	Bissap	2	1	40.99
		...		

Fournisseur

id_fournisseur	compagnie	pays
1	EtsDupleix	France
2	Miamland	France
	...	

Categorie

id_categorie	nom_categorie
1	Boissons
8	Confiserie
	...

Exprimez en algèbre relationnelle les requêtes suivantes (pour simplifier on supposera les noms uniques).
Réponse : La correction ne contient pas toutes les corrections possibles même dans le cas où plusieurs solutions sont proposées. Dans la plupart des questions, il existe beaucoup de requêtes possibles.

1. Quels sont les fournisseurs répertoriés ?
   Réponse : $\Pi_{compagnie}[Fournisseur]$
2. Quels sont les produits coûtant plus de 15 euros, en affichant le nom du produit et le pays des fournisseur où l'on peut le commander.
   Réponse : $\Pi_{nom\_produit,pays}[\sigma_{prix\_unitaire>15}[Fournisseurs \bowtie Produit]]$
3. Quels sont les fournisseurs qui fournissent au moins 2 produits ?
   Réponse : On le fait en plusieurs étapes. Déjà on est intéressé par les tables `Fournisseur` et `Produit`. On commence par retirer toutes les informations non pertinentes de `Produit` :
   $R_1 = \pi_{nom\_produit,id\_fournisseur}[Produit]$

   À partir de là, pour avoir deux produits distincts sur l'identifiant :
   $R_2 = \pi_{id\_fournisseur}\left[\sigma_{nom\_produit \neq np2}\left[[R_1] \bowtie \rho_{nom\_produit \to np2}[R_1]\right]\right]$
   Maintenant on peut conclure pour les compagnies : $R_3 = \pi_{compagnie}\left[[R_2] \bowtie [Fournisseur]\right]$
   Autre solution :

   $R'_3 = \pi_{compagnie}\left[\sigma_{nom\_produit \neq np2}\left[[R_1] \bowtie \rho_{nom\_produit \to np2}[R_1] \bowtie [Fournisseur]\right]\right]$
   On rappelle que les noms de produits sont supposés uniques, comme leur id. C'est ce qui nous permet de travailler avec les noms de produits (si les noms de produits n'étaient pas uniques et qu'on souhaitait obtenir les fournisseurs proposant 2 produits que ces produits aient le même non ou pas, il aurait fallu utiliser id_produit au lieu de nom_produit).
4. Quels sont les fournisseurs qui fournissent des Boissons mais pas de Confiserie ?
   Réponse : On remarque déjà que c'est une soustraction. Donc on cherche les fournisseurs qui font des boissons, et on leur retire ce qui font des confiseries. Pour avoir un fournisseur qui offre une catégorie donnée, il faut faire une jointure des trois tables :
   soit $R_1 = Produit \bowtie Fournisseur \bowtie Categorie$.
   Dès lors, une solution est :
   $\Pi_{compagnie}[\sigma_{nom\_categorie='Boissons'}[R_1]] - \Pi_{compagnie}[\sigma_{nom\_categorie='Confiserie'}[R_1]]$

5. Réfléchir aux requêtes suivantes : lesquelles sont exprimables en algèbre relationnelle (en se limitant à la version classique étudiée en cours) et lesquelles ne le sont pas ?
   (a) les produits les moins chers (tous les produits renvoyés ont donc par conséquent le même prix).
   (b) les fournisseurs qui vendent exactement 3 boissons différentes.
   (c) les fournisseurs qui vendent au plus 6 boissons différentes.
   (d) les fournisseurs qui vendent autant de boissons que de confiseries

   Réponse :

   (a) exprimable : on commence par calculer le prix minimal, en soustrayant de l'ensemble des prix ceux qui ne sont pas minimals (un prix n'est pas minimum s'il existe un prix plus faible). Puis on renvoye par jointure les produits associés à ce prix minimal.
   (b) exprimable : ce sont les fournisseurs vendant au moins 3 boissons différentes auxquels on soustrait ceux qui en vendent au moins 4.
   (c) exprimable : tous les fournisseurs auxquels on soustrait ceux qui vendent au moins 7 boissons différentes.
   (d) non exprimable : informellement, cela impose de compter et on ne peut pas le faire en algèbre relationnelle, ce qu'on admettra ici : ce n'est pas une preuve formelle et je ne m'aventurerai pas à en tenter une. Ce genre de propriétés se montre généralement comme suit (lorsque l'on ne trouve pas de meilleure astuce) : on montre que toute formule d'algèbre relationnelle de taille x peut se traduire en une formule logique de taille f(x) sur les relations ...Pour simplifier on peut considérer la formule comme booléenne. Puis on montre que pour toute formule logique, il existe deux instances (on choisira en général de grandes instances parce qu'une formule de taille k est capable de compter "un peu", mais seulement jusqu'à une valeur qui dépend de la taille de la formule) l'une satisfaisant la formule, l'autre pas, telles que la formule logique sera incapable de distinguer ces deux instances.

Exercice 6 : Estimer la cardinalité du résultat. (30 mins)

Soient R_1 et R_2 deux relations de schéma respectifs $R_1(a, b)$ et $R_2(c, b)$, et de cardinalités respectives c_1 et c_2.

Question : quelle sont la cardinalité maximale et minimale des expressions ci-dessous :

(1) $\pi_a(R_1)$
(2) $\sigma_{b=5}(R_1)$
(3) $R_1 \bowtie R_2$

Question : même question si b est une clef primaire de R_1 ? Et si en plus $R_2.b$ est une clef étrangère vers $R_1.b$?

Réponse :

1. plus exactement : $\min(1, c_1) \leq c \leq c_1$ (la cardinalité minimale est 1, ou 0 si $c_1 = 0$). Clés ne changent rien. Remarque : R_2 clé étrangère ne donne aucune info sur R_1.
2. $0 \leq c \leq c_1$. Si clé, $0 \leq c \leq 1$.
3. $0 \leq c \leq c_1 \times c_2$. Si $R_1.b$ clé primaire : $0 \leq c \leq c_2$. Si $R_2.b$ clé étrangère : $c = c_2$

Exercice 7 : SQL (surtout agrégation) (60 mins)

Considérons le schéma ci dessous pour une base de données représentant les notes de la promo Peip0 de l'école en 2030.

ETUDIANT(id INT, nom Varchar2(10), prenom Varchar2(10), age INT).
RESULTAT(idEtudiant INT, idExamen INT, note INT).

NB. 1) idÉtudiant est bien entendu une clef étrangère 2) on suppose que les tables d'entrée ne contiennent pas de doublons. Votre résultat ne doit alors pas en avoir non plus. 3) on suppose une contrainte d'unicité sur (idEtudiant, idExamen).

1. Écrivez une requête qui renvoie les résultats (nom, prénom et note) des étudiants pour l'examen (`idExamen`) 54.
2. Écrivez une requête permettant au directeur des études de connaitre l'âge du plus jeune de ses grimauds d'école.
3. . . .et le nombre d'étudiants de chaque âge
4. Le professeur Corneille se demande dans quelle mesure la célèbre réplique de Rodrigue (*Le Cid II,2*) se vérifie dans les résultats à sa dernière interro surprise (idExamen=270). Vous devez donc écrire une requête affichant la note moyenne et la note maximale pour chaque âge.
5. Proposez une requête SQL calculant la plus mauvaise note parmi toutes celles répertoriées.
6. Le proviseur du lycée souhaite avoir les nom et prénom de tous les cancres ayant plus de 20 notes en-dessous de 8.
7. Donner une requête renvoyant le nom et prénom de tous les étudiants qui ont au moins une note deux fois plus basse (au moins) que leur note moyenne.

Réponse :

```
/* ---------- requete 1 ---------- */

SELECT nom, prenom, note
FROM Etudiant JOIN Resultat ON Resultat.idEtudiant = Etudiant.id
WHERE idExamen = 54;

/* ---------- requete 2 ---------- */

SELECT MIN(age)
FROM Etudiant;


/* ---------- requete 3 ---------- */

SELECT COUNT(*)
FROM Etudiant
GROUP BY age;

/* requete un peu ambigue: peut-etre :
SELECT age, COUNT(*)
FROM Etudiant
GROUP BY age;
*/

/* ---------- requete 4 ---------- */
SELECT AVG(note), MAX(note)
FROM Etudiant
        JOIN Resultat ON Resultat.idEtudiant = Etudiant.id
WHERE idExamen=270
```

```
GROUP BY age;


/* ---------------- requete 5 ---------------- */

SELECT MIN(note)
FROM Resultat;


/* ---------------- requete 6 ---------------- */
SELECT DISTINCT nom, prenom
FROM Etudiant, Resultat
WHERE Etudiant.id=Resultat.idEtudiant
AND Etudiant.note <=8
GROUP BY nom, prenom, id
HAVING COUNT(*)>20;

/* ---------------- requete 7 ---------------- */
SELECT DISTINCT nom, prenom
FROM Etudiant, Resultat
WHERE Etudiant.id=Resultat.idEtudiant
GROUP BY nom, prenom, id
HAVING MIN(note)<0.5*AVG(note);
;
```

Exercice 8 : Schéma (20 mins)

On considère les 3 schéma suivants : (idp représente l' identifiant du produit, idc : id du constructeur, dp : description du produit, nc nom du constructeur), mais cela ne devrait pas influencer votre réponse.

1. Identifier le schéma relationnel le plus adapté pour enregistrer les informations en évitant au maximum les redondances dans chaque cas répertorié à droite (nombre max de constructeur par produit et vice versa).
2. Donner une requête SQL qui renvoie les produits commençant par *A* du producteur *Martin* sur chacun des schémas.

1. P(idp, idc np, dp, nc).
2. P'(idp, idc np, dp, nc).
3. T1(idp, np, dp, idc) T2(idc, nc)
4. S1(idp, np, dp) S2(idc, nc, idp)
5. R1(idp, np, dp) R2(idc, nc), R3(idp, idc)

nb prod. par c.	**nb const. par p.**
max : 1	max 1
max : ∞	max : ∞
max : ∞	max :1
max : 1	max : ∞

Réponse :

nb prod. par c.	**nb const. par p.**	**solution**
max : 1	max 1	P
max : ∞	max : ∞	R1, R2, R3 (mieux que P' pour éviter les redondanc
max : ∞	max :1	T1, T2
max : 1	max : ∞	S1, S2