

# Module 5b – Exercices et problèmes

Cours IGE487 – Modélisation de bases de données

Chargé de cours Tudor Antohi

# Objectifs

- Solutions Devoir 1
- Commentaires Devoir 2
- Exercices AR
- Exercices DF
- Exercices DMV
- Exercices Normalisation
- Calcule de couts
- Optimisations de SQLs

# **Solutions Devoir 1**

# Sujet 1 AR

1. Considérez une base de données avec le schéma suivant :

**Personne** : (nom, âge, sexe) est une clé

**Fréquentation** (nom, restaurant) : (nom, restaurant) est une clé

**Mange** (nom, repas) : (nom, repas) est une clé

**Sert** (restaurant, repas, prix) : (restaurant, repas) est une clé

Écrivez des expressions d'algèbre relationnelle pour les neuf requêtes suivantes. (Attention : certaines des requêtes vers la fin sont un peu difficiles.)

# Sujet 1 AR - clarifications

Observations

**Personne** : toutes les personnes

**Fréquentation (nom, restaurant)** : les restaurants fréquentés par les personnes

**Mange (nom, repas)** : tous les repas mangés par les personnes, n'importe où, à la maison ou dans les restaurants, ou dans les visites

**Sert (restaurant, repas, prix)** : les repas servis dans les restaurants

Écrivez des expressions d'algèbre relationnelle pour les neuf requêtes suivantes. (Attention : certaines des requêtes vers la fin sont un peu difficiles.)

# Sujet 1 AR (1)

a. Retrouvez toutes les restaurants fréquentées par au moins une personne de moins de 18 ans.

$\pi_{\text{restaurant}} (\sigma_{\text{age} < 18} \text{Personne} \bowtie \text{Fréquentation})$

b. Trouvez les noms de toutes les femmes qui mangent de repas aux viandes ou au salade (ou les deux).

$\pi_{\text{nom}} (\sigma_{\text{sexe} = 'F' \text{ AND } (\text{repas} = \text{'viande'} \text{ OR } \text{repas} = \text{'salade'})} (\text{Personne} \bowtie \text{Mange}))$

# Sujet 1 AR (2)

c. Trouvez les noms de toutes les femmes qui mangent à la fois de la repas aux viandes et au salade.

$\pi_{\text{nom}} (\sigma_{\text{sexe}='F' \text{ AND } (\text{repas} = \text{'viande'})}) (\text{Personne} \bowtie \text{Mange})$

$\cap \pi_{\text{nom}} (\sigma_{\text{sexe}='F' \text{ AND } (\text{repas} = \text{'salade'})}) (\text{Personne} \bowtie \text{Mange})$

d. Trouvez tous les restaurants qui servent au moins un repas qu'Amélie mange pour moins de 10 \$.

$\pi_{\text{restaurant}} (\sigma_{\text{nom}='Amélie'} \text{Mange} \bowtie \sigma_{\text{prix} < 10} \text{Sert})$

# Sujet 1 AR (3)

e. Trouvez toutes les restaurants fréquentées uniquement par des femmes ou uniquement par des hommes.

$$\begin{aligned} & (\pi_{\text{restaurant}} (\sigma_{\text{sexe}='F'} \text{ Personne} \bowtie \text{Frequentation})) \\ & - \pi_{\text{restaurant}} (\sigma_{\text{sexe}='M'} \text{ Personne} \bowtie \text{Frequentation})) \\ & \cup \\ & (\pi_{\text{restaurant}} (\sigma_{\text{sexe}='M'} \text{ Personne} \bowtie \text{Frequentation})) \\ & - \pi_{\text{restaurant}} (\sigma_{\text{sexe}='F'} \text{ Personne} \bowtie \text{Frequentation})) \end{aligned}$$



# Devoir 1 Sujet 3 (1)

Temp (regionID, name, high, low)

Region (RegionId)

$$T1(rID, h) = \pi_{regionID, high} Temp$$

$$T2(rID, h) = \pi_{regionID, low} Temp$$

Deux projections créent deux relations, une avec les highs de température , une autre avec les lows.

# Devoir 1 Sujet 3 (2)

Temp (regionID, name, high, low)

Region (RegionId)

$$T3(regionID) = \pi_{rID}(T1 \bowtie_{h < high} Temp)$$

$$T4(regionID) = \pi_{rID}(T2 \bowtie_{l > low} Temp)$$

Deux jointures qui projettent :

T3 avec les regions avec les temperatures plus petites que le maximum

T4 avec les regions avec les temperatures plus grandes que le minimum.

# Devoir 1 Sujet 3 (3)

Temp (regionID, name, high, low)

Region (RegionId)

$$T5(regionID) = \pi_{regionID} Temp - T3$$

$$T6(regionID) = \pi_{regionID} Temp - T4$$

$$Result(n) = \pi_{name}(Temp \bowtie (T5 \cup T6))$$

Deux jointures qui projettent :

- T5 avec la region qui a le maximum de temperature (exclusion)
- T6 avec la region qui a le minimum de temperature (exclusion)

Résultat final : **les noms des regions qui ont le maximum et(ou) le minimum de temperature.**  
**Combient de tuples ?**

# **Commentaires Devoir 2**

# Pourquoi il-existe du 5FN ? (1)

Une hypothèse est que 5FN c'est absolument inutile.

<https://stackoverflow.com/questions/37660527/when-to-use-the-fifth-normal-form-5nf>

Mais faire attention aux gens qui font des déclarations formidables sans ajouter les mots : *parce que*. Always question that authority.

Commençons avec Wikipedia.

[https://en.wikipedia.org/wiki/Fifth\\_normal\\_form](https://en.wikipedia.org/wiki/Fifth_normal_form)

On normalise en 5NF et on crée à partir d'une table 4FN de 33 cellules , 3 tables qui comptent un total de 48 cellules.

# Pourquoi existe du 5FN ? (2)

La redondance est au rendez-vous et le prix est plus de stockage...

Mais dans la réalité on peut avoir 100 vendeurs et pour chaque vendeur, 10 brands en moyenne et pour chaque brand 10 produits qu'on vend en moyenne.

On part de 4FN avec une table qui  $100 \times 10 \times 10 \times 3 = 30000$  cellules

Si on est en 5FN , on a  $2 \times 100 \times 10 + 2 \times 100 \times 10 + 2 \times 10 \times 10 = 4200$  cellules.

On commence a réduire les couts de stockage pour une bonne redondance aussi.

# Pourquoi existe-il du 6FN ?

<u>Deviceld</u>	Timestamp	Température	Illumination
1	1	20	11
1	2	20	12
1	3	20	13
1	4	20	14
1	5	19	15
1	6	19	16
1	7	19	17

On réduit la quantité de donnée en faisant simplement.

±

<u>Deviceld</u>	Timestamp	Température
1	1	20
1	2	19

□

<u>Deviceld</u>	Timestamp	Illumination
1	1	11
1	2	12
1	3	13
1	4	14
1	5	15
1	6	16
1	7	17

On sait que aucun changement dans température depuis le timestamp 1. A longue terme on gagne en terme d'espace aussi.

Cependant l'illumination bougent plus vite.

Les protocoles de communication doivent communiquer moins de données.

Transmettre seulement les différences fait du sense aussi dans le stockage.

On sépare dans des tables indépendants de mesure.

# 4FN vers 5FN décomposition

Impossible de décomposer 4FN en 5FN sans créer le 6FN.

1. **4FN** permet de projections sur des champs qui ne sont pas des clés, et ne permet pas de dépendances multivaleur.
2. **5FN** permet de projections de jointure non-triviales sur les clés.
3. **6FN** ne permet que de projections de jointures triviales sur les clés

Une relation en 4FN qui n'est pas en 5FN ou 6FN est seulement une table sans attributs non-premiers. Tous les attributs font partie de la clé. La décomposition génère seulement 4FN similaires ou directement 6FN avec une seule clé primaire.



# Exercices Algèbre Relationnelle

# Q1

**R(A,C) et S(B,C,D) - quel tuple existe en  $R \bowtie S$**

A	C
3	3
6	4
2	3
3	5
7	1

B	C	D
5	1	6
1	5	8
4	3	9

1. 6,4,3,9
- 2. 7,5,1,6**
3. 3,5,1,6
4. 2,3,1,6

**Solution :**

C'est une jointure, la colonne commune est C.

La jointure entre le tuple numéro 5 de R et le tuple numéro 1 de S se trouve dans la solution.

La jointure contient numéro 2.

## Q2

**$R(A,B)$  et  $S(B,C,D)$  - quel tuple existe en  $R \bowtie_C S$  et la condition est  $R.B = S.B$  and  $R.A < S.C$**

A	B
1	a
7	t
2	g
4	c
9	t

B	C	D
c	5	6
a	7	8
t	8	9

1. (4, c, c, 5, 6)
2. (2, g, g, 7, 8)
3. (2, g, t, 8, 9)
4. (1, a, a, 8, 9)

**Solution :**

C'est une jointure theta qui combine un produit cartésien avec une restriction, bref on va avoir 5 colonnes :  **$R.a$ ,  $R.b$ ,  $S.b$ ,  $S.c$ ,  $S.d$ .**

Il existe une solution ou  **$R.b = S.b$  et  $R.a < S.c$**

# Q3

**R(A,B,C) et S(A,B,C) - quel tuple existe en  $(R - S) \cup (S - R)$**

A	B	C
1	2	3
4	2	3
4	5	6
2	5	3
1	2	6

A	B	C
2	5	3
2	5	4
4	5	6
1	2	3
2	5	3

## Solution

Le résultat sont les tuples qui existent en R et n'existent pas en S et les tuples qui existent en S et n'existent pas en R.

4,5,6 existent dans les deux.

4,5,3 n'existe pas

**4,2,3 existe en R et pas en S**

1,2,3 existe dans les deux

1. (4, 5,6)
2. (4,5,3)
3. **(4,2,3)**
4. (1,2,3)

# Q4

$R(A,B)$  et  $S(B,C)$  - tuples avec des valeurs INT.

a.  $\pi_{A,C}(R \bowtie \sigma_{B=1} S)$

b.  $\pi_A(\sigma_{B=1} R) \times \pi_C(\sigma_{B=1} S)$

c.  $\pi_{A,C}(\pi_A R \times \sigma_{B=1} S)$

Deux expressions produisent le même résultat. Quelle expression produit le résultat différent ?

**Solution :** La bonne réponse est C. La projection de C va contenir de tuples de R avec les valeurs B différents de 1.

# Exercices Dépendance Fonctionnelle

# Quiz 1 - DF

On considère une relation  $R(A,B,C,D,E)$  avec des dépendances fonctionnelles suivante :

$AB \rightarrow C, C \rightarrow D, BD \rightarrow E$

Quel ensemble d'attributs parmi les suivants **ne** détermine **pas** fonctionnellement E ?

1. BCD

2. **AD**

3. BC

4. ABC

## **Solution**

BCD détermine E parce que BD détermine E

**AD ne déterminé pas E , aucun chemin existe vers E a partir de A seulement ou AC ou C seulement.**

BC implique E parce que C implique D et que BD détermine E

ABC implique E parce que BC implique E comme démontré.

# Quiz 2 - DF

1) On considère une relation  $R(A,B,C,D,E)$  avec des dépendances fonctionnelles suivantes :

$D \rightarrow C$ ,  $CE \rightarrow A$ ,  $D \rightarrow A$ ,  $AE \rightarrow D$

Quel choix parmi les suivantes est une clé ?

1. BCE

2. A

3. AD

4. CDE

## **Solution**

Une clé doit déterminer les attributs non-premiers.

B n'est pas déterminé par aucun attribut donc B doit faire partie de clé.

Le candidat est BCE.

BCE détermine D parce que CE détermine A et AE détermine D.

BCE détermine A parce que CE détermine A.

Bref BCE est une clé parce que BCE détermine les autres attributs A et D



# Quiz 3 - DF

Supposons une relation  $R(A,B,C,D)$  ayant les dépendances fonctionnelles suivantes :

$$A \rightarrow B, B \rightarrow C, C \rightarrow A$$

On appelle l'ensemble de dépendance ci-dessus,  $S1$ . Un autre ensemble de dépendances fonctionnelles  $S2$  est équivalent au  $S1$ , s'ils ont exactement les mêmes DFs issues de  $S1$  et  $S2$ .

Quel ensemble de DFs est équivalent à l'ensemble  $S1$  défini ci-dessus ?

1.  $A \rightarrow BC, C \rightarrow AB$
2.  **$A \rightarrow BC, B \rightarrow AC, C \rightarrow AB$**
3.  $A \rightarrow B, B \rightarrow C, C \rightarrow B$
4.  $A \rightarrow BC, B \rightarrow AC$

## Solution

1. Clairement non, impossible de déduire si  $B$  détermine  $C$
2. **Oui,  $A$  détermine  $B$  et  $C$ , donc  $A$  détermine  $B$ . La même chose pour les autres dépendances fonctionnelles.**
3.  $A$  n'est pas déterminé.
4. Il manque la détermination de  $A$  par  $C$ .

# Quiz 4 - DF

Supposons une relation  $R(A,B,C)$  n'ayant pour l'instant qu'un tuple  $(0,0,0)$ , et qui satisfait toujours les dépendances fonctionnelles suivantes :

- $A \rightarrow B$  et  $B \rightarrow C$ .

Lequel des tuples suivants peut être inséré dans  $R$  ?

1.  **$(1,1,0)$**

## **Solution**

On sait que une valeur 0 de  $A$  détermine une valeur 0 de  $B$ .

2.  $(0,2,0)$

On sait aussi que une valeur 0 de  $B$  détermine une valeur 0 de  $C$

3.  $(0,1,1)$

$(1,1,0)$  peut être une solution parce que on a aucune violation. Une valeur 1 de  $A$  peut impliquer une valeur 1 de  $B$  et une valeur 1 de  $B$  peut impliquer une valeur 0 de  $C$ .

4.  $(1,0,1)$

$(0,2,0)$  et  $(0,1,1)$  violent  $A \rightarrow B$  parce que une valeur 0 de  $A$  détermine 0 en  $B$ .

$(1,0,1)$  violent  $B \rightarrow C$  parce que une valeur 0 de  $B$  détermine 0 en  $C$ .

# **Exercices Dépendances Multi-Valeur**

# Q1 – DMV

Nous avons une représentation (instance) d'une relation  $R(A,B,C)$  suivante :

Quelle dépendance multi-valeur parmi les suivantes **n'est pas** satisfaite par cette instance de  $R$  ?

1.  $C \twoheadrightarrow A$
2.  $AB \twoheadrightarrow A$
3.  $BC \twoheadrightarrow C$
4.  $B \twoheadrightarrow C$

A	B	C
1	2	3
1	3	2
1	2	2
3	2	1
3	2	3

## Solution

$AB \twoheadrightarrow A$  et  $BC \twoheadrightarrow C$  sont triviales.

Une dépendance multi-valeur dans une variable relation est testées en effectuant de swaps sur les multi-valeurs détectées et confirmer que le tuple résultant existe déjà.

Dans notre cas  $C \twoheadrightarrow A$  on a (3,1) et (3,3). Si on swap (C,A) les tuples 1 et 5 on arrive aux même valeurs. (1,2,3) et (3,2,3) qui existent.

Par contre  $B \twoheadrightarrow C$  peut créer des valeur différentes. Si on swap (B,C) avec les valeurs (2,3) et (2,1) entre les tuples 1 et 4 ca nous donnent un tuple (1,2,1) qui n'existent pas dans la relation initiale. Donc  $B \twoheadrightarrow C$  ne répond pas aux besoins DMV.

# Q2 – DMV

Nous avons une représentation (instance) d'une relation  $R(A,B,C,D)$  suivante :

Quelle dépendance multivaluée parmi les suivantes **est** satisfaite par cette instance de  $R$  ?

1.  $B \twoheadrightarrow C$
2.  $BD \twoheadrightarrow C$
3.  $B \twoheadrightarrow CD$
4.  **$C \twoheadrightarrow B$**

A	B	C	D
1	2	3	4
1	3	3	3
1	3	3	4
1	2	3	3
2	2	4	4
2	4	2	4
2	4	4	4
2	2	2	4

## Solution

$B \twoheadrightarrow C$  tombe quand le swap est appliqué entre les tuples 1 et 5.  
Un nouvel tuple qui n'existe pas (1,2,4,4) est créé.

$BD \twoheadrightarrow C$  tombe quand le swap est appliqué entre les tuples 1 et 5.  
Le swap de (2,3  $\twoheadrightarrow$  4) et (2,4  $\twoheadrightarrow$  4) crée le aussi le (1,2,4,4).  
 $B \twoheadrightarrow CD$  tombe aussi quand le swap est appliqué entre les tuples 1 et 5.

Le swap de (2  $\twoheadrightarrow$  3,4) et (2  $\twoheadrightarrow$  4,4) crée le aussi le (1,2,4,4).

Il reste  **$C \twoheadrightarrow B$** . Ex.: Si on teste le swap avec (3  $\twoheadrightarrow$  2 et 3  $\twoheadrightarrow$  3) on est correcte sur les tuples 1 et 3.

# Q3 – DMV – Solution

Nous avons une instance de la relation  $R(A,B,C,D,E)$  :

On considère les dépendances multivaleur suivantes :

(1)  $A \twoheadrightarrow B$

(2)  $B \twoheadrightarrow D$

Si  $R$  contient  $(0,1,2,3,4)$  et  $(0,4,6,7,8)$  quel autre tuple existe en  $R$  :

1.  $(0,1,5,7,4)$

2.  $(0,1,2,3,8)$

3.  **$(0,1,6,3,8)$**

4.  $(0,1,6,3,4)$

**Solution:**

$A \twoheadrightarrow B$  si on swap  $(0,1)$  et  $(0,4)$  on obtient  $(0,1,6,7,8)$  et  $(0,4,2,3,4)$

$B \twoheadrightarrow D$  si on swap  $(1,3)$  et  $(4,7)$  on obtient  $(0,5,2,7,8)$  et  **$(0,1,6,3,8)$**

**Autres tuples ne correspondent a part de  $(0,1,6,3,8)$ .**

# Exercises Normalisation

# Clôture d'une relation – algorithme

La clôture d'une relation pour un ensemble des attributs et l'ensemble des attributs qui ont une DF de la clé.

- Ajouter les attributs contenus dans l'ensemble d'attributs  $X$  à l'ensemble de résultats  $X^+$ .
- Ajouter à l'ensemble de résultats  $X^+$  les attributs qui peuvent être déterminés fonctionnellement à partir des attributs déjà contenus dans l'ensemble de résultats.
- Répéter l'étape 2 jusqu'à ce que aucun attribut ne puisse être ajouté à l'ensemble de résultats  $X^+$ .



# Clôture d'une relation – exemple

On nous donne la relation  $R(A, B, C, D, E)$ . Cela signifie que le tableau  $R$  a cinq colonnes :  $A, B, C, D$  et  $E$ . On nous donne également l'ensemble des dépendances fonctionnelles :  $\{A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow E\}$ .

## Solution

On commence avec  $A$ . Nous ajoutons  $A$  à  $\{A\}^+$ .

Quelles colonnes peuvent être déterminées étant donné  $A$  ? Nous avons  $A \rightarrow B$ , donc nous pouvons déterminer  $B$ . Par conséquent,  $\{A\}^+$  est maintenant  $\{A, B\}$ .

Quelles colonnes peuvent être déterminées à partir de  $A$  et  $B$  ? Nous avons  $B \rightarrow C$  dans les dépendances fonctionnelles, donc nous pouvons déterminer  $C$ . Par conséquent,  $\{A\}^+$  est maintenant  $\{A, B, C\}$ .

Maintenant, nous avons  $A, B$ , et  $C$ . Quelles autres colonnes pouvons-nous déterminer ? Eh bien, nous avons  $C \rightarrow D$ , donc nous pouvons ajouter  $D$  à  $\{A\}^+$ .

Maintenant, on a  $A, B, C$ , et  $D$ . On peut ajouter autre chose ? Oui, puisque  $D \rightarrow E$ , nous pouvons ajouter  $E$  à  $\{A\}^+$ .

Nous avons utilisé toutes les colonnes de  $R$  et nous avons utilisé toutes les dépendances fonctionnelles.  $\{A\}^+ = \{A, B, C, D, E\}$ .  $A$  détermine tous les attributs, donc  $A$  est une clé.

**Obs.:** on n'est pas en FNBC dans ce cas !

# Q1 – Normalisation

On considère une relation  $R(A,B,C,D,E)$  ayant les **dépendances multivaleur** suivantes :

$A \twoheadrightarrow B$ ,  $B \twoheadrightarrow D$  et **sans dépendance fonctionnelle**.

On suppose qu'on décompose  $R$  en 4FN. On peut obtenir différentes décompositions à la fin, selon l'ordre de traitement des violations en 4FN.

Quel schéma de relation parmi les suivants pourrait être dans la décomposition **finale** en 4FN ?

1. ACDE

2. **ACE**

3. CE

4. ABCE

## **Solution:**

On suit les règles de décomposition.

$(A,B,C,D,E)$  est décomposé en  $(A,B)$  et  $(A,C,D,E)$  pour séparer la première dépendance multivaleur.

Dans la 2-ème étape, la règle de transitivité nous donne  $(A \twoheadrightarrow D)$  et on sépare  $(A,C,D,E)$  en  $(A,D)$  et  **$(A,C,E)$**

# Q2 – Normalisation

On considère une relation  $R(A,B,C,D,E)$  en FNBC. Si  $(A,B,C)$  est la clé unique de  $R$  quelle DF est possible dans FNBC ?

1.  $BCE \rightarrow A$

**2.  $ABC \rightarrow D$**

3.  $BCDE \rightarrow A$

4.  $ACE \rightarrow D$

## **Solution**

Une relation est en FNBC si pour chaque DF non triviale, les attributs du côté gauche contiennent une clé.

Dans le cas 1,3,4 c'est pas possible, aucune ensemble des attributs a gauche ne sont pas de clés a cause de  $E$  ou  $D$ .

Par contre dans le cas de 2 , on a une clé  $(A,B,C)$  qui détermine  $D$  et répond a FNBC.

# Q3 – Normalisation

On considère une relation  $R(A,B,C,D)$ . Pour quel ensemble de DFs (dépendances fonctionnelles)  $R$  est en FNBC ?

1.  $A \rightarrow D, C \rightarrow A, D \rightarrow B, AC \rightarrow B$
2.  $C \rightarrow D, CD \rightarrow A, AB \rightarrow C, BD \rightarrow A$
- 3.  $C \rightarrow B, D \rightarrow A, C \rightarrow D, A \rightarrow C$**
4.  $ABE \rightarrow D, BCD \rightarrow A, D \rightarrow C, ACD \rightarrow B$

**Dans le cas 1 :** La clôture de  $A$  est  $(A,D,B)$ . La clôture de  $C$  est  $(C,A,B,D)$ . Mais  $A$  détermine  $D$  donc  $A$  doit faire partie de la clé. Dans ce cas la clé  $(C,A)$  ne respecte pas le FNBC parce que  $A$  détermine  $C$ . etc.

## Solution

Une relation est en BCNF si pour chaque FD non trivial, les attributs du côté gauche contiennent une clé.

Pour vérifier si un ensemble d'attributs  $S$  contient une clé, on calcule la clôture des attributs dans  $S$  en utilisant tous les FD. Si la fermeture est constituée de tous les attributs de la relation, alors les attributs contiennent une clé, sinon non.

**Dans le cas 3,** si on commence avec  $\{C\}^+ = \{C\}$ , on ajoute  $B$ , et on a  $\{C\}^+ = \{B,C\}$  on ajoute  $A$  et on a à la fin  $\{C\}^+ = \{B,C,D,A\}$ .

$D$  aussi  $\{D\}^+ = \{B,C,D,A\}$ .  $A$  aussi a une clôture  $\{A\}^+ = \{B,C,D,A\}$ .

$C$  ne peut pas être une clé unique pour FNBC parce que  $D$  détermine  $A$ .  $D$  doit faire partie de la clé et la clé unique est  $(A, C, D)$  et répond aux besoins de FNBC, n'importe quelle FD contient à droite une clé.

Subtile ;-)

# Q4 – Normalisation

On considère une relation  $R(A,B,C,D,E)$ . Pour quel ensemble de DFs (dépendances fonctionnelles)  $R$  est en FNBC ?

1.  **$BCD \rightarrow E, BDE \rightarrow C, BE \rightarrow D, BE \rightarrow A$**
2.  $AD \rightarrow B, ABC \rightarrow E, BD \rightarrow A, B \rightarrow A$
3.  $BE \rightarrow D, B \rightarrow E, D \rightarrow E, CD \rightarrow A$
4.  $ABD \rightarrow C, ACD \rightarrow E, ACE \rightarrow B, BC \rightarrow E$

# Q5 – Normalisation

On considère une relation  $R(A,B,C,D)$  ayant les dépendances fonctionnelles et multivaleur suivantes :

$A \rightarrow B$ ,  $C \rightarrow D$ ,  $B \twoheadrightarrow C$

On suppose qu'on décompose  $R$  en 4FN. On peut obtenir différentes décompositions à la fin, selon l'ordre de traitement des violations en 4FN. Quel schéma de relation parmi les suivants pourrait être dans la décomposition finale en 4FN ?

1. ABCD

2. BC

3. ABD

4. BCD

**Solution:**

En 4FN on est en FNBC sans DMV.

AC est une clé minimale qui détermine B et D.

La transitivité donne  $A \twoheadrightarrow C$  et  $A \twoheadrightarrow D$ .

Évidemment ABCD, ABD et BCD contiennent de dépendances multi-valeurs.

Par contre BD peut exister dans plusieurs étapes de décomposition

1. Première décomposition est (A,B) et (B,C,D) suivie par la décomposition finale de (B,C,D) en

2. (A,B), **(B,C)**, (B,D)

# Exercices Calcule de Cout

# Coûts logiques – nombre d'accès blocs

## Informations de catalogue utilisées dans les fonctions de coût

- Informations sur la taille d'un fichier (**grandeur**)
- nombre d'enregistrements (tuples) (**r**),
- taille d'enregistrement (**R**),
- nombre de blocs physiques (**b**)
- facteur de blocage (**bfr**), nombre des valeurs dans un bloc physique

## Informations sur les index et les attributs d'indexation d'un fichier

- Nombre de niveaux (**x**) de chaque index multiniveau
- Nombre de blocs d'index de premier niveau (**bl1**)
- Nombre de valeurs distinctes (**d**) d'un attribut
- Sélectivité (**sel**) d'un attribut
- Cardinalité (**s ou card**) de sélection d'un attribut. ( $s = sel * r$ )



# Exemples de calculs logiques de couts

La table EMPLOYEE a  $rE = 10\,000$  enregistrements stockés dans  $bE = 2\,000$  blocs de disque avec un facteur de blocage  $bfrE = 5$  enregistrements/bloc.

1. **index cluster sur Salary**, niveaux B-tree  $xSalary = 3$  et une cardinalité moyenne  $cardSalary = 20$ .  $selSalary = 20/10000 = 0,002$
2. **index secondaire sur l'attribut clé Ssn**, avec  $xSsn = 4$  ( $cardSsn = 1$ ,  $selSsn = 0,0001$ ).
3. **index secondaire sur l'attribut non clé Dno**, avec  $xDno = 2$  et de premier niveau blocs d'index  $bl1Dno = 4$ .  $NDV(Dno, EMPLOYEE) = 125$  valeurs distinctes pour Dno, la sélectivité est  $selDno = (1/NDV(Dno, EMPLOYEE)) = 0,008$ , et la cardinalité est  $cardDno = (rE * selDno) = (rE/NDV(Dno, EMPLOYEE)) = 80$ .
4. **index secondaire sur Sexe**, avec  $xSex = 1$ . Il y en a  $NDV(Sex, EMPLOYEE) = 2$  valeurs pour l'attribut Sexe, la cardinalité est  $selSex = (rE/NDV(Sex, EMPLOYEE)) = 5000$ .

# Exemples de calculs de couts (1)

$\sigma_{Dno=5 \text{ AND SALARY}>30000 \text{ AND Sex='F'}}(\text{EMPLOYEE})$

1. Estimation de coût pour Dno=5 :

$$\text{CS6a} = \text{xDNO} + 1 + \text{cardinalité de DNO} = 2 + 1 + 80 = 83$$

Les accès :

- a) Un parcours en profondeur sur l'arbre B-Tree pour trouver la valeur d'index , valeur 2
- b) Accès dans le bloc
- c) 80 accès pour la cardinalité moyenne de DNO, un accès pour chaque valeur existante.

# Exemples de calculs de couts (2)

$\sigma_{Dno=5 \text{ AND SALARY}>30000 \text{ AND Sexe='F'}}$  (EMPLOYEE)

2.  $CS4 = x + (b/2) \Rightarrow \text{Salaire} > 30000$  donne une estimation de coût :

$$CS4 = x_{\text{Salaire}} + (b_E/2) = 3 + (2000/2) = 1003$$

Les accès :

- a) Un parcours en profondeur sur le index cluster qui contient aussi les valeurs d'enregistrement sur la première valeur de 30000 si le cas.
- b) 1000 accès pour la cardinalité moyenne de DNO, le nombre de blocs en accès séquentiel estimé sur tous les blocs de fichier qui n'est pas ordonné. Pourquoi, parce que 30000 peut être la valeur minimale de salaire dans le index cluster ;-)

# Exemples de calculs de couts (3)

$\sigma_{Dno=5 \text{ AND SALARY}>30000 \text{ AND Sex='F'}}(\text{EMPLOYEE})$

3. CS6a = x + 1 + cardinalité de sexe  $\Rightarrow$  L'utilisation de la condition (Sexe = 'F') donne une estimation de coût sur le index B-Tree :

$$\text{CS6a} = x_{\text{Sexe}} + \text{cardinalité}_{\text{Sexe}} = 1 + 5000 = 5001$$

# En réalité les estimations d'un plan sont complexes

- ✓ Volume de IO
  - ✓ Temps CPU
  - ✓ Nombre de rangées
  - ✓ Nombre de lectures physiques sur disques
  - ✓ Nombre de lectures de mémoire
  - ✓ Temps d'exécution estimé
- + statistiques d'exécution qui serviront pour les algorithmes adaptives

# Oracle

Id	Operation	Name	Rows (Estim)	Cost	Time Active(s)	Start Active	Execs	Rows (Actual)	Read Reqs	Read Bytes	Cell Offload	Mem (Max)	Activity (%)	Activity Detail (# samples)
0	SELECT STATEMENT				12086	+5	1	864K					0.01	Cpu (1)
1	CONCATENATION				12086	+5	1	864K						
2	FILTER				12191	+4	1	864K					0.03	Cpu (4)
3	FILTER				12191	+4	1	26M					0.01	Cpu (1)
4	NESTED LOOPS		241	251K	12191	+4	1	26M					0.02	Cpu (3)
5	NESTED LOOPS		241	251K	12191	+4	1	26M					0.07	Cpu (8)
6	NESTED LOOPS		241	251K	12232	+4	1	26M					0.05	Cpu (6)
7	NESTED LOOPS		5407	233K	12242	+4	1	86M						
8	MERGE JOIN CARTESIAN		1	35	12242	+4	1	1000						
9	TABLE ACCESS BY INDEX ROWID	REF1	1	3	1	+4	1	1				104K		
10	INDEX RANGE SCAN	REF1_PK	1	2	12242	+4	1	1						
11	BUFFER SORT		84	32	12242	+4	1	1000						
12	TABLE ACCESS BY INDEX ROWID	STAGE	84	32	1	+4	1	1000						
13	INDEX RANGE SCAN	STAGE_IDX1	84	4	1	+4	1	1000						
14	PARTITION RANGE ITERATOR		8292	232K	12232	+4	1000	86M						
15	TABLE ACCESS STORAGE FULL	TAB1	8292	232K	12245	+1	1000	86M	103M	521GB	1.96%	7M	51.81	gc buffer busy acquire (1) latch: cache buffers chains (1) Cpu (1196) gcs drmm freeze in enter server mode (2) reliable message (5) cell single block physical read (2827) cell smart table scan (1977) read by other session (304)
16	PARTITION RANGE ITERATOR		1	12	12191	+4	86M	26M					0.42	Cpu (51)
17	TABLE ACCESS BY LOCAL INDEX ROWID	TAB2	1	12	12191	+4	86M	26M	4M	28GB			32.14	gc cr grant 2-way (20) gc cr request (2) gc remaster (6) Cpu (319) gcs drmm freeze in enter server mode (4) latch: gc element (2) cell single block physical read (3563) Cpu (292) cell single block physical read (1557) Cpu (20) cell single block physical read (1)
18	INDEX RANGE SCAN	TAB2_IX1	166	3	12210	+2	86M	26M	1M	11GB			15.17	
19	INDEX UNIQUE SCAN	MTD_PK	1	1	12242	+4	26M	26M	292	2MB			0.17	Cpu (20) cell single block physical read (1)
20	TABLE ACCESS BY INDEX ROWID	REF2	1	2	12191	+4	26M	26M	7	57344			0.11	
21	TABLE ACCESS BY INDEX ROWID	CONTROLTAB	1	1	1	+4	1	1						
22	INDEX UNIQUE SCAN	CONTROLTAB_PK	1	1	1	+4	1	1						
23	MINUS				102	+4	25	3						
24	TABLE ACCESS BY INDEX ROWID	CUST_ORG_PK	1	3	942	+4	25	10						
25	INDEX UNIQUE SCAN	MC_PK	1	2	942	+4	25	25						
26	SORT UNIQUE NOSORT		1	4	8	+4	25	9						
27	TABLE ACCESS BY INDEX ROWID	REF1	1	3	8	+4	25	9						
28	INDEX RANGE SCAN	REF1_PK	1	2	8	+4	25	9						

Predicate Information (identified by operation id):

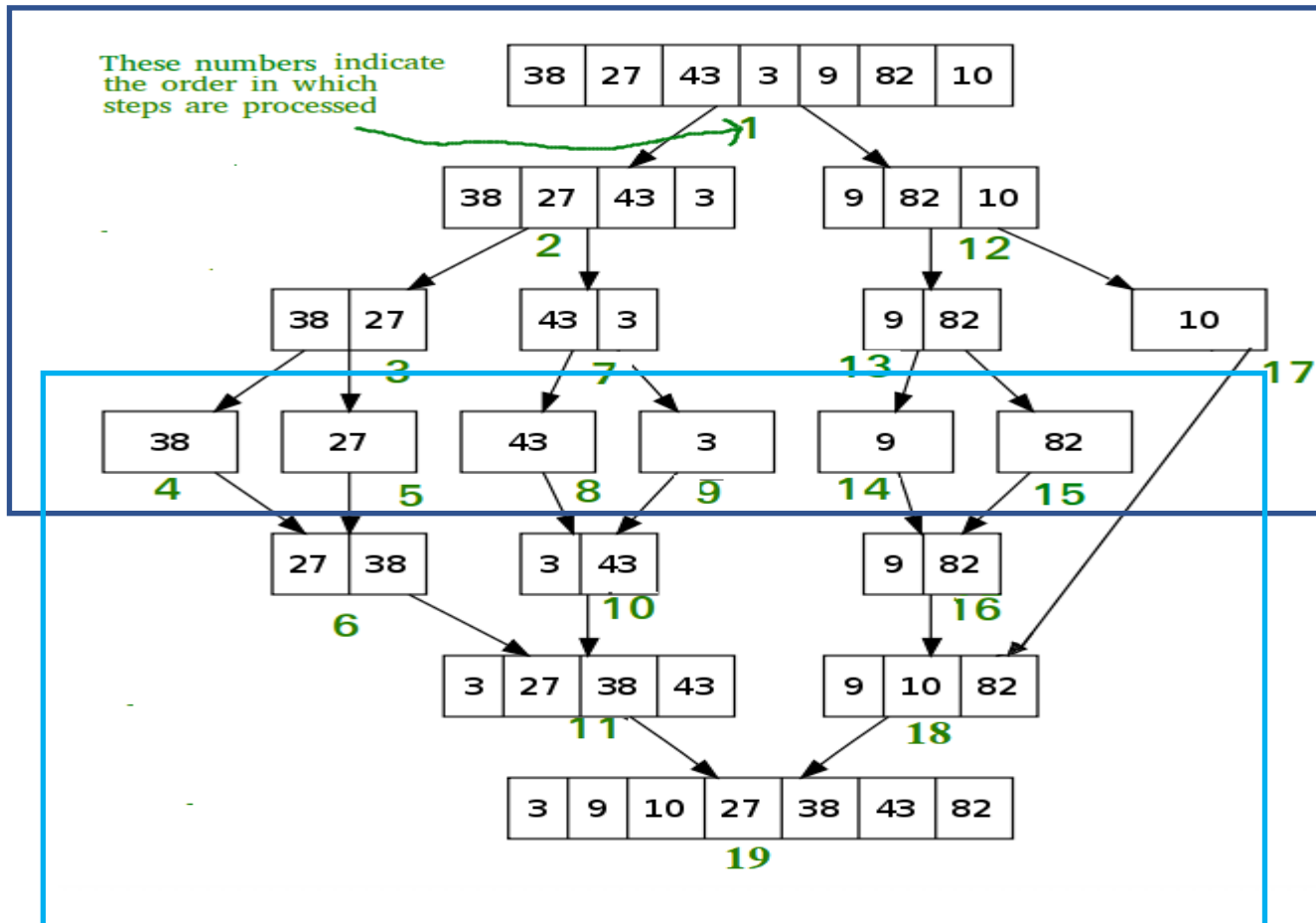
```

2 - filter( EXISTS ( (SELECT /*+ INDEX_RS_ASC ("CUST_ORG_PK" "MC_PK") */ "CUST_ID" FROM "USER1"."CUST_ORG_PK" "CUST_ORG_PK"
WHERE "CUST_ID"=:B1 AND "CUST_ORG_PK"."INDICATR"=:Y) MINUS (SELECT /*+ INDEX_RS_ASC ("REF1" "REF1_PK") */
TO_NUMBER("VAL") FROM "USER1"."REF1" "REF1" WHERE "PUSER"=:ZZZ AND "EDATE" .ge. TRUNC(SYSDATE@!) AND TO_NUMBER("VAL")=:B2
AND "SDATE" .le. TRUNC(SYSDATE@!)))
3 - filter( EXISTS (SELECT /*+ INDEX_RS_ASC ("CONTROLTAB" "CONTROLTAB_PK") */ 0 FROM "USER2"."CONTROLTAB" "CONTROLTAB" WHERE
"CONTROLTAB"."CNTRLID"=9999 AND NVL("CONTROLTAB"."STATUS",'F')='S'))
9 - filter("REF1"."EDATE" .ge. TRUNC(SYSDATE@!))
10 - access("REF1"."PUSER"='XXX' AND "REF1"."NAME"='CODE' AND "REF1"."SDATE" .le. TRUNC(SYSDATE@!))
13 - access("STAGE"."NAME"='XXX' AND "STAGE"."STAT"='I')

```

# Algorithmes Tri : SORT-MERGE

Sort-Merge est l'algorithme typique , basé sur le principe de *diviser-et- régner*



**Phase de tri :** les exécutions du fichier pouvant tenir dans la mémoire tampon sont triées et réécrites sur le disque en tant qu'exécutions triées temporaires

$$nR = \lceil b/nB \rceil + 1$$

( $nR$  is #executions,  $nB$  nombre de buffers,  $b$  = #blocs)

$b = 1024$ ,  $nB = 5$ ,

**Fusion (Merge):** les passages triés sont fusionnés au cours d'une ou plusieurs passes.

$dM$  = # pistes pouvant être fusionnées en un seul passage.

$dM = \min(nB-1, nR)$

#passes =  $\text{int}(\log_{dM}(nR)) + 1$

#accès blocs =  $(2*b) + (2*(b*(\log_{dM} b)))$

# Questions 1 – tri-fusion + solution

**Un fichier de 4 096 blocs doit être trié avec un espace tampon disponible de 64 blocs. Combien de passes seront nécessaires dans la phase de fusion (merge) de l'algorithme de tri-fusion (sort merge) externe ?**

## **Solution :**

Les exécutions triées sont fusionnées lors d'un ou plusieurs passages.

Le degré de fusion (dM) est le nombre de passages qui peuvent être fusionnés à chaque passe.

À chaque passe, un bloc est nécessaire pour chacune des exécutions fusionnées, et un bloc est nécessaire pour contenir le bloc du résultat de la fusion.

- $nR = b/nb = 4096 / 64 = 64$  exécutions et 64 fichiers triés de 64 blocs produits.
- $dM = \min\{nB - 1, nR\} = 63$  fichiers au même temps

La première passe produit un fichier de  $63 \times 64$  blocs = 4032 blocs et un autre de 64 blocs.

La deuxième passe prend le fichier de 64 blocs de la première passe et fait la fusion avec le fichier de 4032 blocs.

Résultat final = 2 passes.

La formule

- $\log_{dM}(nR) = \text{ceil}(\log_{63}(64)) = 2$  passes



# Questions 2

**Un indice non dense peut-il être utilisé par un opérateur d'agrégation ? Pourquoi ou pourquoi pas ? Illustrez par un exemple.**

**Solution:**

**Non-dense index, Sparse index** – contient des enregistrements d'index uniquement pour certaines valeurs de clé de recherche. C'est applicable lorsque les enregistrements sont triés séquentiellement sur la clé de recherche.

Tout comme les index de livres, les *index sparse* pointent vers de pages mais ne contiennent pas du contenu..

Un index non dense a :

- Une entrée pour chaque bloc
- Une seule entrée pour chaque valeur différente, mais pas pour tous les valeurs !

Les opérateurs d'agrégation MAX, AVERAGE, SUM et COUNT ne peuvent pas être implémentés avec un index non dense, car toutes les valeurs de données ne sont pas visibles dans l'index.

Un index non dense peut être utilisé pour implémenter la fonction MIN, si dans le bloc de données, le pointeur de clé d'index pointe vers la plus petite valeur de clé.

Seulement l'index dense peut être utilisé pour implémenter les fonctions d'agrégation comme AVG et SUM, car dans l'index dense, chaque enregistrement a une entrée d'index associée.

# Ordre de jointures – a discuter

Example: List the name of all customers who have ordered a product from a supplier located in Davis.

$$\pi_{CName}(\sigma_{SAddress \text{ like } '%Davis\%'}(SUPPLIERS \bowtie ORDERS \bowtie CUSTOMERS))$$

# PostgreSQL création de tables

- **CREATE TABLE** public.demotable (
  - num **numeric NULL**,
  - id **int4 NULL**,
  - id1 **int4 NULL**
- **);**

**1 million rangées**

**SET max\_parallel\_workers\_per\_gather = 0 ou 2 au besoin.**

**CREATE INDEX demoidx ON public.demotable USING btree (num);**

# PostgreSQL avec full table scan sans indexes

```
explain analyze SELECT * FROM demotable WHERE num = 21000;
```

Results 1



Execution plan - 1



Execution plan - 2

explain analyze SELECT \* FROM demo |  Enter a SQL expression to filter results (use Ctrl+Space)



ABC QUERY PLAN




1	Seq Scan on demotable (cost=0.00..18870.00 rows=1 width=19) (actual time=313.129..313.129 rows=0)
2	Filter: (num = '21000'::numeric)
3	Rows Removed by Filter: 1000000
4	Planning Time: 0.072 ms
5	Execution Time: 313.198 ms

# PostgreSQL avec full table scan et parallelisme

```
SET max_parallel_workers_per_gather = 2;
```

```
explain analyze SELECT * FROM demotable WHERE num = 21000;
```

	ABC QUERY PLAN
1	Gather (cost=1000.00..12578.43 rows=1 width=19) (actual time=131.004..132.815 rows=0 loops=1)
2	Workers Planned: 2
3	Workers Launched: 2
4	-> Parallel Seq Scan on demotable (cost=0.00..11578.33 rows=1 width=19) (actual time=127.012..127.012 rows=0 loops=3)
5	Filter: (num = '21000'::numeric)
6	Rows Removed by Filter: 333333
7	Planning Time: 0.271 ms
8	Execution Time: 132.841 ms

# PostgreSQL avec indexe , clause égalité

```
explain analyze SELECT * FROM demotable WHERE num = 210;
```

ults 1



Execution plan - 1



Execution plan - 2

lain analyze SELECT \* FROM demo



Enter a SQL expression to filter results (use Ctrl+Space)



ABC QUERY PLAN



Index Scan using demoidx on demotable (cost=0.42..8.44 rows=1 width=19) (actual time=0.065..0.066 rows=0 loops=1)

Index Cond: (num = '210'::numeric)

Planning Time: 0.175 ms

Execution Time: 0.086 ms

# PostgreSQL avec un index , clause inégalité

```
explain analyze SELECT * FROM demotable WHERE num > 210;
```

results 1



Execution plan - 1



Execution plan - 2

explain analyze SELECT \* FROM demo



Enter a SQL expression to filter results (use Ctrl+Space)



ABC QUERY PLAN



1	Seq Scan on demotable (cost=0.00..18870.00 rows=787140 width=19) (actual time=0.045..205.625 rows=789663 loops=1)
2	Filter: (num > '210'::numeric)
3	Rows Removed by Filter: 210337
4	Planning Time: 0.082 ms
5	Execution Time: 237.324 ms

# PostgreSQL avec deux indexes

```
CREATE INDEX demoidx1 ON public.demotable USING btree (id);
```

```
CREATE INDEX demoidx2 ON public.demotable USING btree (id1);
```

```
explain analyze SELECT * FROM demotable WHERE id = 32768 and id1 = 100111
```

Results 1



Execution plan - 1



Execution plan - 2

explain analyze SELECT \* FROM demo Enter a SQL expression to filter results (use Ctrl+Space)

	ABC QUERY PLAN
1	Index Scan using demoidx2 on demotable (cost=0.42..8.45 rows=1 width=19) (actual time=0.092..0.092 rows=0 loops=1)
2	Index Cond: (id1 = 100111)
3	Filter: (id = 32768)
4	Rows Removed by Filter: 1
5	Planning Time: 0.088 ms
6	Execution Time: 0.111 ms



# PostgreSQL avec un index composé

```
CREATE INDEX demoidx3 ON public.demotable USING btree (id,id1);
```

```
explain analyze SELECT * FROM demotable WHERE id = 32768 and id1 = 100111
```

Its 1



Execution plan - 1



Execution plan - 2

ain analyze SELECT \* FROM demo |  Enter a SQL expression to filter results (use Ctrl+Space) 

**ABC** QUERY PLAN



Index Scan using demoidx3 on demotable (cost=0.42..8.45 rows=1 width=19) (actual time=0.041..0.041 rows=0 loops=1)

Index Cond: ((id = 32768) AND (id1 = 100111))

Planning Time: 0.256 ms

Execution Time: 0.057 ms

# PostgreSQL avec un indexe hash

```
create index idxhash_numbers_numbercol on demotable using hash (id1);
```

```
explain analyze SELECT * FROM demotable WHERE id1 = 32768  
create index idxhash_numbers_numbercol on demotable using hash (id1);
```

Results 1 × Execution plan - 1 Execution plan - 2

explain analyze SELECT \* FROM demo Enter a SQL expression to filter results (use Ctrl+Space)

ABC	QUERY PLAN
1	Index Scan using idxhash_numbers_numbercol on demotable (cost=0.00..8.02 rows=1 width=19) (actual time=0.014..0.015 rows=1 loops=1)
2	Index Cond: (id1 = 32768)
3	Planning Time: 0.190 ms
4	Execution Time: 0.037 ms

# PostgreSQL avec un indexe brin

```
create index idxhash_numbers_numbercol on demotable using brin (num);
```

```
explain analyze SELECT * FROM demotable WHERE num = 32768
```

Results 1



Execution plan - 1



Execution plan - 2

explain analyze SELECT \* FROM demo



Enter a SQL expression to filter results (use Ctrl+Space)



ABC QUERY PLAN



1	Index Scan using demoidx on demotable (cost=0.42..8.44 rows=1 width=19) (actual time=0.015..0.015 rows=0 loops=1)
2	Index Cond: (num = '32768'::numeric)
3	Planning Time: 0.235 ms
4	Execution Time: 0.033 ms

**PostgreSQL avec un indexe gin, gist et partitionnement - a venir dans la présentation de bases de données temporelles.**

# SÉRIALISATION

# Recouvrement d'une cédule seriale

**Memo :** *La condition d'un ordonnancement récupérable (récouvrable) est la suivante : un ordonnancement  $S$  est récupérable si aucune transaction  $T$  dans  $S$  n'est validée (COMMIT) tant que toutes les transactions  $T'$  qui ont écrit un élément  $X$  avant que  $T$  lit n'ont pas été validées. On écrit :*

- $r1(X)$  = lire1(X) = lire dans la transaction T1
- $w1(X)$  = écrire1(X) = écrire dans la transaction T1

Les ordonnancements (cédules, calendriers) suivants sont-ils recouvrables ?

**Sa:**  $r1(X); r2(X); w1(X); r1(Y); w2(X); c2; w1(Y); c1;$

**Réponse : oui**, aucun incident possible d'écriture en T1(2) et lecture après en T1(1)

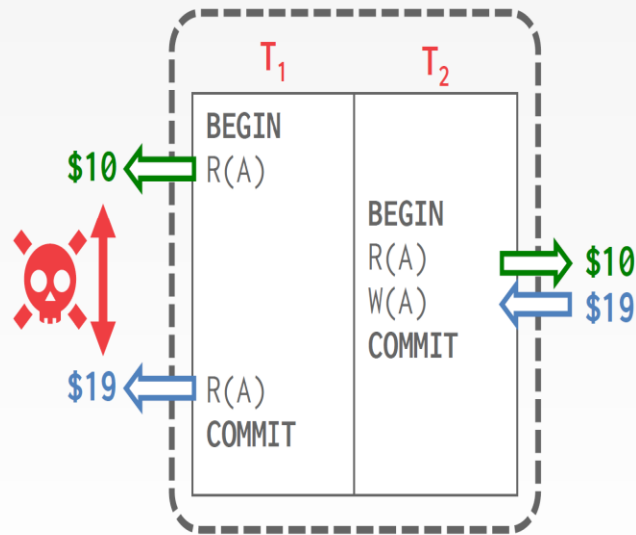
**Sc:**  $r1(X); w1(X); r2(X); r1(Y); w2(X); c2; a1;$

**Réponse : non**, T2 effectue la lecture de X après l'écriture de X en T1 et T2 est validée avant T1.

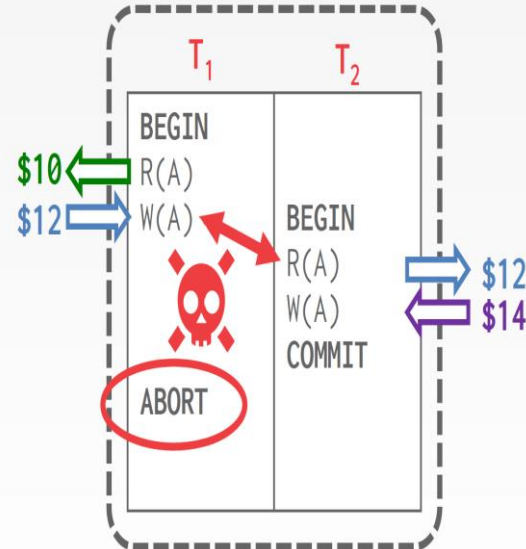
# Conflits de sérialisation - revue

En utilisant la notion d'équivalence de conflit, nous définissons un ordonnancement  $S$  comme étant sérialisable s'il est c-équivalent à un ordonnancement sériel  $S'$ . Conflits :  $RW$  (*Lecture - écriture*),  $WR$  (*écriture-lecture*),  $WW$  (*écriture-écriture*)

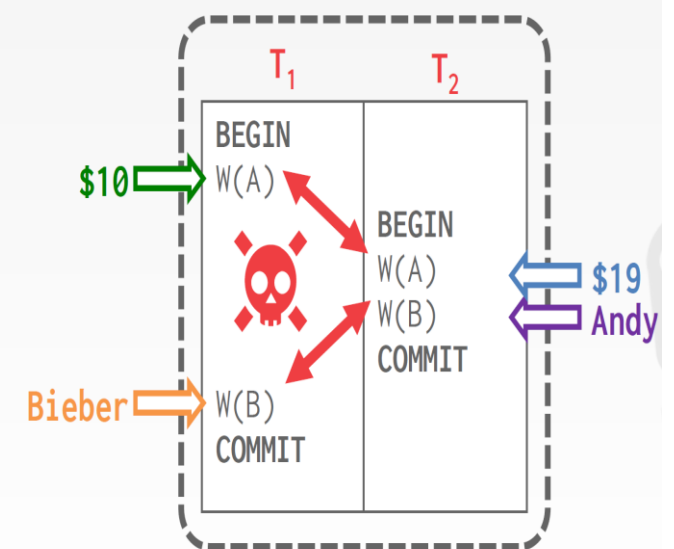
Unrepeatable Reads



Reading Uncommitted Data ("Dirty Reads")



Overwriting Uncommitted Data



# Graph de sérialisation - exemple

Arrêt de  $T_i$  à  $T_j$  si :

- Une opération  $O_i$  de  $T_i$  est en conflit avec une opération  $O_j$  de  $T_j$  et
- $O_i$  apparaît plus tôt dans le programme que  $O_j$ .

**Question S :**  $r1(x) \ r1(y) \ w2(x) \ w1(x) \ r2(y)$  est-il serialisable ?

- $r1(x)$  arrive avant  $w2(x)$ , arrêt de  $T_1$  vers  $T_2$
- $w2(x) \ w1(x)$  un autres arrêt  $T_2$  vers  $T_1$

**Réponse : NON** , graph cyclique





**ACID**

# Valeurs finale selon l'isolation

- Svp. comprendre bien utilisant deux transactions communes :
  - Read Uncommitted
  - Read Committed
  - Snapshot Isolation
  - Serializable
  - Repeatable Read