

Corrigé Examen Administration des SGBD
M1 : IL-SII

Enoncé :

Un centre de recherche sur le médicament et l'innovation thérapeutique rassemble plusieurs chercheurs de disciplines diverses pour concevoir des médicaments du futur au sein d'un ensemble de laboratoires spécialisés. Chaque laboratoire dispose d'équipes de chercheurs permanents et associés travaillant sur des projets innovants.

Le schéma relationnel suivant modélise les activités de ce centre :

Laboratoire (**Nomlab**, thématique, nbr-equipes)

Equipe (**Codequipe**, Nomequipe, *Nomlab*)

Chercheur (**Codechercheur**, Nomchercheur, adresse, spécialité, catégorie, *Codequipe*)

Projet (**Codeprojet**, domaine, type, phase, *Codequipe*)

Affectation_Projet (**Codechercheur**, **Codeprojet**, datedeb, datefin, pourcentage)

En gras et souligné les clés, en italique les clés étrangères

Nous disposons les informations suivantes

| | |
|--|-------------------------------|
| Laboratoire | Card(Laboratoire)=8 |
| Equipe | Card(Equipe)=32 |
| Chercheur | Card(Chercheur)= 160 |
| Projet | Card(Projet)=320 |
| Affectation_Projet | Card(Affectation_Projet)=1280 |
| La page d'entrée/Sortie i.e. page d'échange entre la mémoire centrale et secondaire peut contenir 8 enregistrements (tuples). On suppose que chaque attribut est codé sur 50 octets. | |

- Les thématiques appartiennent à la liste : Cancers, maladies infectieuses, maladies cardiovasculaires, système immunitaire.
- Les spécialités des chercheurs appartiennent à la liste : biologistes, chimistes, experts médecins, experts pharmaciens.
- La catégorie de chercheurs est soit permanent, soit associé, soit vacataire
- Le domaine du projet appartient à la liste :
 - cibles thérapeutiques : identifier et valider de nouvelles cibles thérapeutiques
 - résistances : nouvelles stratégies contre la réduction d'efficacité des médicaments
 - adressage : développement de nouveaux systèmes d'adressage basés sur la micro et nano technologie.
- Le type de projet est soit innovant, soit industriel
- Les phases du projet sont : étude académique, étude pré clinique, étude clinique. Les projets en phase clinique terminale vont pouvoir alimenter les sociétés de biotechnologies qui ne font pas l'objet de cette étude.

Exercice 1 : Fonctions générales de SGBD et Optimisation (11.5 points)

1. La base de données est accédée par plusieurs utilisateurs de différentes catégories : **(a) un administrateur** qui a tous les privilèges possibles et propriétaire des tables du schéma relationnel précédent, **(b) un directeur de centre** qui a une vue de synthèse de données et accède en lecture à toutes les tables de la BD, **(c) des directeurs de laboratoire** qui consultent, modifient et suppriment les équipes de leur laboratoire, **(d) des chefs d'équipe** qui affichent modifient et suppriment les chercheurs, les projets, les affectations de leur équipe, **(e) les chercheurs** qui consultent les projets et les affectations de projets de leur équipe. Un directeur de laboratoire possède tous les privilèges de chef d'équipe.

a. Donner la meilleure stratégie d'attribution des droits d'accès pour cette BD. **0.25 pts**

Pour faciliter la gestion des privilèges : d'aboord, on définit des rôles au nombre des catégories des utilisateurs (**administrateur, directeur de centre, directeur de laboratoire, chef d'équipe et chercheur**). Puis, On attribue à ces rôles les différents privilèges. A la fin, on associe ces rôles aux différents utilisateurs.

- b. Donner les commandes SQL utilisées pour appliquer cette stratégie. 0.5 pts * chaque ligne=2.5 pts

| Définition de rôle | Attribution de privilèges | Création des utilisateurs et association des rôles |
|--------------------------------|--|--|
| Create role adiminstrateur ; | Grant all privileges to adiminstrateur | Create user adimn identified by psw; Grant adiminstrateur to admin; |
| Create role directeur_centre ; | Grant select on Laboratoire to directeur_centre ; Grant select on Equipe to directeur_centre ; Grant select on Chercheur to directeur_centre ; Grant select on Projet to directeur_centre ; Grant select on Affectation_Projet to directeur_centre ; | Create user directeurcentre identified by psw; Grant directeur_centre to directeurcentre; |
| Create role chef_equipe ; | Grant select on Equipe to chef_equipe ; Grant select, update, delete on Chercheur to chef_equipe ; Grant select, update, delete on Projet to chef_equipe ; Grant select, update, delete on Affectation_Projet to chef_equipe ; | Create user chefequipe identified by psw; Grant chef_equipe to chefequipe ; |
| Create role directeur_labo ; | Grant select on Laboratoire ; Grant select, update, delete on Equipe to directeur_labo ; Grant chef_equipe to directeur_labo | Create user directeurolabo identified by psw; Grant directeur_labo to directeurolabo; |
| Create role chercheur ; | Grant select on Equipe to chercheur ; Grant select on Chercheur to chercheur ; Grant select on Projet to chercheur ; Grant select on Affectation_Projet to chercheur ; | Create user chercheur identified by psw; Grant chercheur to chercheur; |

- c. Donner le script SQL pour créer les tables Chercheur et Affectation_Projet. Avec toutes les contraintes d'intégrité. 0.5 pts * chaque ligne=1 pts

| |
|--|
| <pre> create table Chercheur (Codechercheur varchar2(10), nomchercheur varchar2(50), adresse varchar2(100), specialite varchar2(50), categorie varchar2(50), Codeequipe varchar2(10) ; constraint pk_chercheur primary key(Codechercheur), constraint ck_specialite check (specialite in ('biologistes', 'chimistes', 'experts médecins', 'experts pharmaciens')), constraint ck_categorie check (categorie in ('permanent', 'associé', 'vacataire')), constraint fk_equipe foreign key(Codeequipe) references Equipe (Codeequipe) on delete cascade); </pre> |
| <pre> create table Affectation_Projet (Codechercheur varchar2(10), Codeprojet varchar2(15), datedeb date, datefin date, pourcentage float, constraint pk_Affectation_Projet primary key(Codechercheur, Codeprojet), constraint ck_date check (datedeb<datefin), constraint ck_pourcentage check (pourcentage>0 and pourcentage<=1), constraint fk_chercheur foreign key(Codechercheur) references Chercheur (Codechercheur) on delete cascade, constraint fk_projet foreign key(Codeprojet) references Projet (Codeprojet) on delete cascade,); </pre> |

2. Le directeur de laboratoire veut

- Avoir le nombre de ses équipes sans avoir à le calculer à chaque fois.
- Définir pour chaque équipe son chef.
- Avoir le nombre de chercheurs sans avoir à le calculer à chaque fois.

a. Que faut-il faire ? Donner les commandes SQL associées ainsi que les répercussions sur les catalogues. **0,5 pt ligne 1 + 1 pt par ligne (2 et 3) = 2,5 pts**

| Besoin | Solution | Effet sur les catalogues |
|--|--|--|
| - Avoir le nombre de ses équipes sans avoir à le calculer à chaque fois. | - Un trigger « maj_nb_equipes_trigger » qui met à jour (incrémente ou décrémente) l'attribut nbr_equipes à chaque ajout, modification ou suppression d'une ligne dans la table Equipe. | |
| - Définir pour chaque équipe son chef. | - Ajouter un attribut Chefequipe dans la table Equipe et définir une clé étrangère sur cet attribut vers la table chercheur (codechercheur). - Alter table Equipe add chefequipe varchar2(10); - Alter table Equipe add constraint fk_chef foreign key(Chefequipe) references Chercheur (Codechercheur) on delete cascade; | - Dans le catalogue attribut : l'insertion d'une ligne décrivant l'attribut (Nom attribut : Chefequipe, type : varchar2(10),) - Dans le catalogue relation : Modification de la cardinalité de la relation attribut : card = card + 1 - Modification de la relation Equipe: Degré = Degré + 1, Taille d'un tuple = Taille d'un tuple + Longueur d'un entier, Version = version +1. |
| - Avoir le nombre de chercheurs sans avoir à le calculer à chaque fois. | - Ajouter un attribut nbr_chercheurs dans la table Laboratoire et définir un trigger « maj_nb_chercheur_trigger » qui met à jour (incrémente ou décrémente) l'attribut nbr_chercheurs à chaque ajout, modification ou suppression dans la table chercheur. - Alter table Laboratoire add nbr_chercheurs integer default 0; | - Dans le catalogue attribut : l'insertion d'une ligne décrivant l'attribut (Nom attribut : nbr_projets, type : integer,) - Dans le catalogue relation : Modification de la cardinalité de la relation attribut : card = card + 1 - Modification de la relation Laboratoire: Degré = Degré + 1, Taille d'un tuple = Taille d'un tuple + Longueur d'un entier, Version = version +1. |

- b. Ecrire les scripts permettant de faire automatiquement ces calculs sachant la BD ne contient pas de tuples. **01pts*chaque ligne=02 pts**

| Trigger | Code PL/SQL |
|---------------------------|--|
| maj_nb_equipes_trigger | <pre> create or replace trigger maj_nb_equipes_trigger after insert or delete or update of(Nomlab) on equipe for each ROW BEGIN CASE WHEN INSERTING THEN update Laboratoire set nbr_equipes=nbr_equipes+1 where Nomlab =:new. Nomlab; WHEN DELETING THEN update Laboratoire set nbr_equipes=nbr_equipes-1 where Nomlab =:old. Nomlab; WHEN UPDATING THEN Begin update Laboratoire set nbr_equipes=nbr_equipes-1 where Nomlab =:old. Nomlab; update Laboratoire set nbr_equipes=nbr_equipes+1 where Nomlab =:new. Nomlab; End; END CASE; EXCEPTION WHEN OTHERS then DBMS_OUTPUT.PUT_LINE('error : ' sqlcode ' ' sqlerrm); END; / </pre> |
| maj_nb_chercheurs_trigger | <pre> create or replace trigger maj_nb_chercheurs_trigger after insert or delete or update of(Codeequipe) on chercheur for each ROW DECLARE NewNomlab Laboratoire.Nomlab%type; OldNomlab Laboratoire.Nomlab%type; begin CASE WHEN INSERTING THEN Begin select Nomlab into NewNomlab from Equipe where Codeequipe=:new.codeequipe; update Laboratoire set nbr_chercheurs= nbr_chercheurs +1 where Nomlab = NewNomlab; end; WHEN DELETING THEN Begin select Nomlab into NewNomlab from Equipe where Codeequipe=:old.codeequipe; update Laboratoire set nbr_chercheurs= nbr_chercheurs -1 where Nomlab = NewNomlab; end; WHEN UPDATING THEN Begin select Nomlab into NewNomlab from Equipe where Codeequipe=:new.codeequipe; update Laboratoire set nbr_chercheurs= nbr_chercheurs +1 where Nomlab = NewNomlab; select Nomlab into NewNomlab from Equipe where Codeequipe=:old.codeequipe; </pre> |

| | |
|--|---|
| | <pre> update Laboratoire set nbr_chercheurs= nbr_chercheurs -1 where Nomlab = NewNomlab; End; END CASE; EXCEPTION WHEN OTHERS then DBMS_OUTPUT.PUT_LINE('error : ' sqlcode ' ' sqlerrm); END; / </pre> |
|--|---|

3. Le directeur de centre veut afficher la liste de projets (codeprojet, domaine) en étude pré clinique des laboratoires des maladies infectieuses.

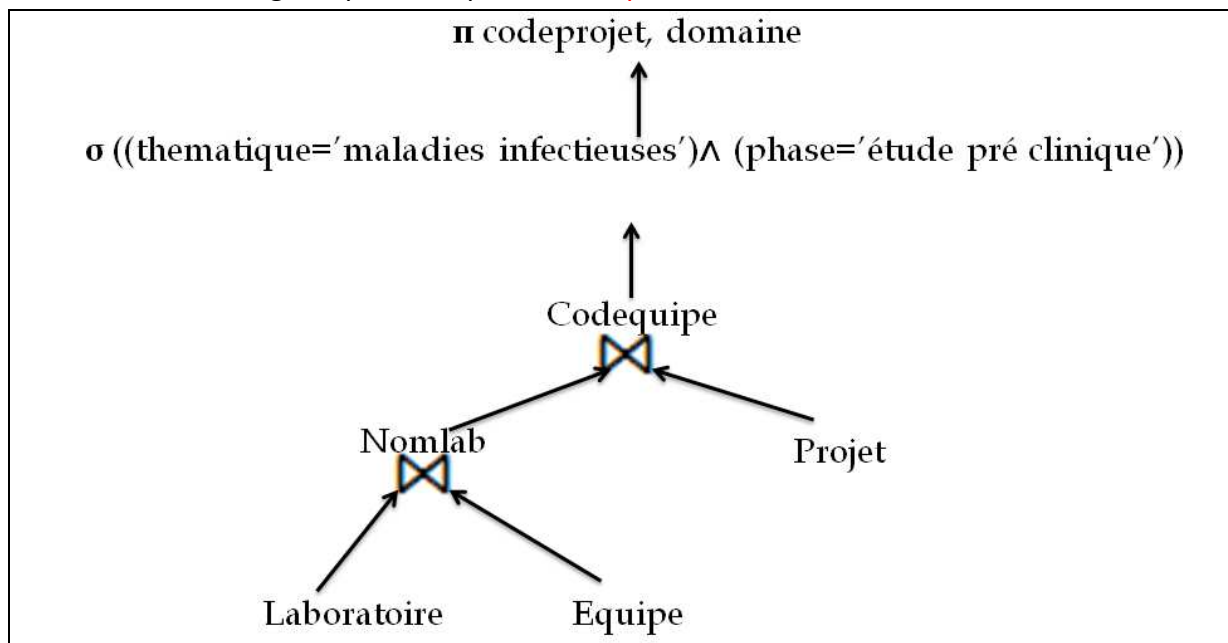
a. Donner la requête SQL. 0.5 pts

```

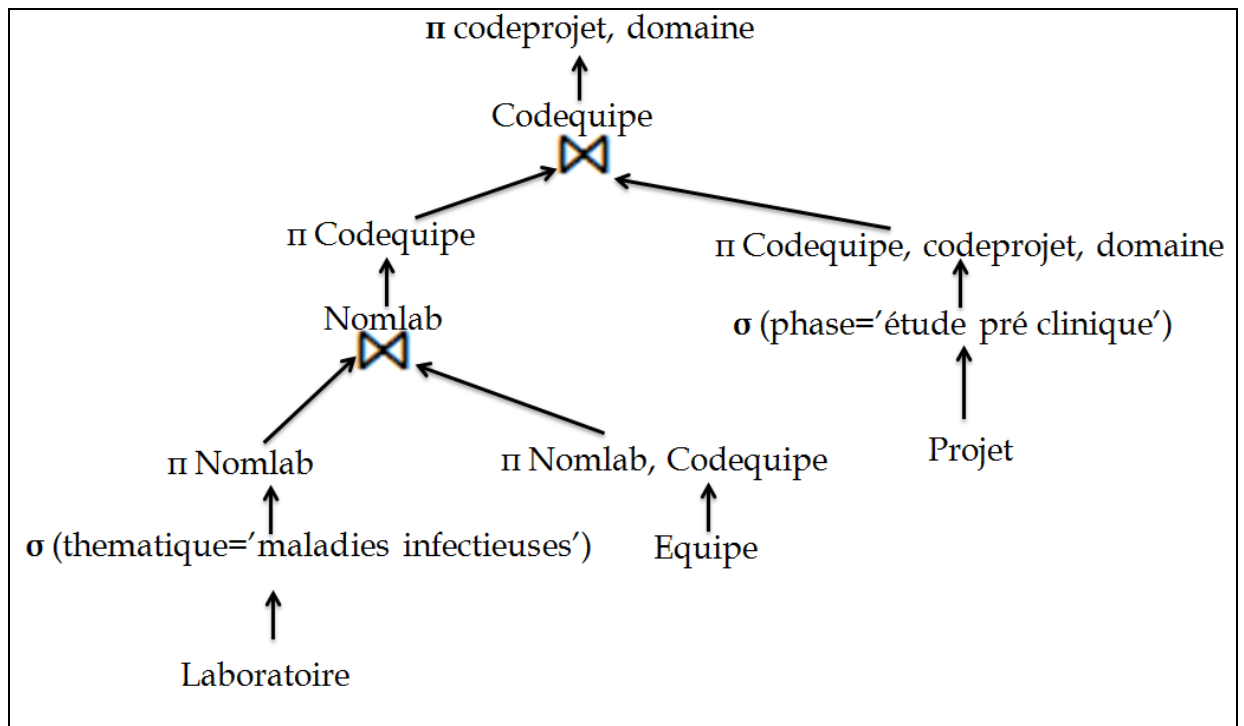
Select codeprojet, domaine
From Laboratoire L, Equipe E, Projet P
Where thematique='maladies infectieuses'
And phase='étude pré clinique'
And L.Nomlab=E.NomLab
And E.Codeequipe=P.Codeequipe;

```

b. Donner un arbre algébrique non optimisé. 0.5 pts



c. Générer un arbre optimisé en utilisant les règles de transformation. 0.5 pts



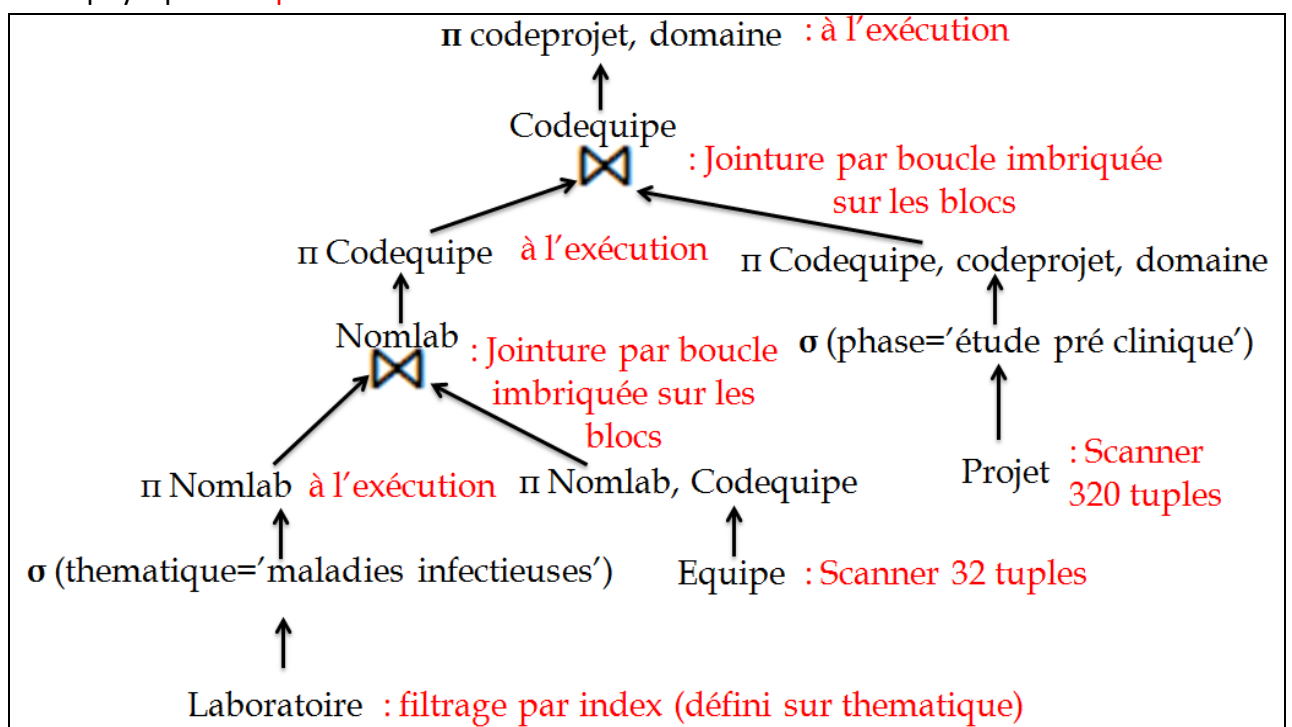
- d. Sachant qu'un index est défini sur l'attribut thématique de la table laboratoire, Quel est le type de cet index. Donner la commande SQL correspondante. **0.5 pts**

L'index est secondaire car il est défini sur un attribut non primaire.

Create index thematique_ix on laboratoire (thematique);

- e. Estimer le coût d'exécution de la requête selon l'arbre optimisé sachant que les résultats intermédiaires peuvent être stockés dans la RAM.

- Plan physique **0.25 pts**



Calcul de coût : **0.25pts* chaque ligne =01 pts**

| Opération | Coût | Taille du résultat |
|--|---|---|
| $R1 = \pi \text{ Nomlab}(\sigma (\text{thematique} = \text{'maladies infectieuses'}) (\text{Laboratoire}))$ | <p>on suppose que toutes les pages index sont en mémoire centrale. L'index qu'on a défini est un index secondaire, la page index contient la clé (thematique) et Nomlab (identifiant de laboratoire), pour la jointure on a besoin que de Nomlab d'où on n'a pas besoin de faire une entrée/sortie.</p> <p>$\text{Card}(\text{Laboratoire}) * (1/\text{Val}(\text{thematique}, \text{Laboratoire}))$ $= 8 * 1/4$ $= 2 \text{ pages}$</p> | <p>$\text{CARD}(R1) = \text{CARD}(\text{Laboratoire}) / \text{Val}(\text{thematique})$ $= 8/4$ $= 2 \text{ tuples}$</p> |
| <p>- Charger la table equipe 2 fois - $R2 = \pi \text{ Nomlab}, \text{Codequipe}(\text{Equipe})$ - Exécuter la jointure : $R3 = \pi \text{ Codequipe} (R1 \bowtie R2)$</p> <p>La jointure sera faite en pipeline car les données peuvent rester en MC (jointure par boucle imbriquée)</p> | <p>$2 * (32/8)$ $= 2 * 4$ $= 8 \text{ pages}$</p> | <p>$\text{CARD}(R2) = 32 \text{ tuples}$ $\text{CARD}(R3)$: Jointure par clé étrangère (Nomlab). nb-tuples = 2 tuples de la table Laboratoire avec 32 tuples de la table Equipe. $32 * 2/8$ D'où 8 tuples comme résultats</p> |
| <p>- Charger la table projet 8 fois $R4 = \pi \text{ Codequipe}, \text{codeprojet}, \text{domaine} (\sigma (\text{phase} = \text{'étude pré clinique'}) (\text{Projet}))$ Exécuter la jointure : $R5 = \pi \text{ codeprojet}, \text{domaine} (R3 \bowtie R4)$</p> <p>La jointure sera faite en pipeline car les données peuvent rester en MC (jointure par boucle imbriquée)</p> | <p>$8 * (320/8)$ $= 8 * 40$ $= 320 \text{ pages}$</p> | <p>$\text{CARD}(R4) =$ $\text{CARD}(\text{projet}) / \text{Val}(\text{phase})$ $= 320 / 4 = 80 \text{ tuples}$ $\text{CARD}(R5)$: Jointure par clé étrangère (Codequipe). nb-tuples = 8 tuples de R3 avec 80 tuples de R4. $8 * 80 / 32$ (Val clé étrangère Codequipe) D'où 20 tuples comme résultats</p> |
| Total | <p>$2 + 8 + 320$ $= 340 \text{ pages}$</p> | <p>$20 * (\text{size}(\text{codeprojet}) + \text{size}(\text{domaine})) \text{ octets}$ $= 20 * (50 + 50) = 2000 \text{ Octes}$</p> |

Exercice 2 : Gestions des accès concurrents et reprise après panne (09.5 points)

1. Si S est un ordonnancement sur les transactions T_1, \dots, T_n , alors le sous-graphe de précédence réduit $\rho(S)$ associé à S est un graphe qui a les transactions dans S comme nœuds, et a une arête de T_i à T_j si et seulement si S contient deux actions conflictuelles $a_i(x)$ dans T_i et $a_j(x)$ dans T_j sur le même élément x tel que $a_i(x)$ précède (pas nécessairement directement) $a_j(x)$ dans S , et il n'y a pas d'action $a_k(x)$ dans S entre $a_i(x)$ et $a_j(x)$, avec k différent de i et j . Prouver ou réfuter les affirmations suivantes en se basant sur la condition suffisante de sérialisabilité du graphe de précédence:

- a. Si $\rho(S)$ est cyclique, alors S n'est pas sérialisable. **01 pts**

L'affirmation peut être prouvée en remarquant que $\rho(S)$ est un sous-ensemble du graphe de précédence $P(S)$ associé à S . En effet, les deux graphes ont évidemment les mêmes nœuds, et, chaque arête dans $\rho(S)$ est le témoin de deux actions conflictuelles dans S , et donc cette arête apparaît aussi dans $P(S)$. d'où si $\rho(S)$ contient un cycle, $P(S)$ l'a aussi.

- b. Si $\rho(S)$ est acyclique, alors S est sérialisable. **01 pts**

L'affirmation peut être réfutée par le contre-exemple S_1 suivant : **R1(X) R2(X) W3(X) W3(Y) W1(Y).**

1. Graphe de précédence de $\rho(S_1)$

Les conflits :

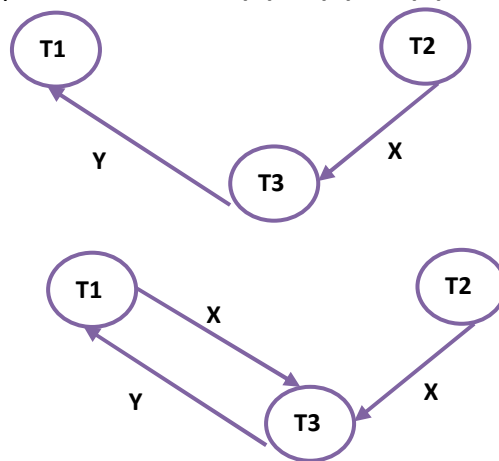
- Sur X : R2(X)-W3(X);
- Sur Y : W3(Y)- W1(Y) ;

2. Graphe de précédence de $P(S_1)$

Les conflits :

- Sur X : R1(X)-W3(X); R2(X)-W3(X);
- Sur Y : W3(Y)- W1(Y) ;

$\rho(S_1)$ est acyclique mais $P(S_1)$ est cyclique.



2. Considérons l'ordonnancement O des transactions T_1, T_2, T_3 et T_4 initialement arrivées aux temps 10, 20, 30 et 40 respectivement.

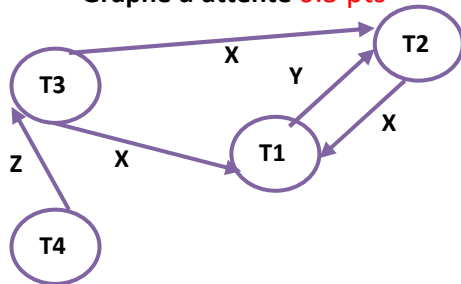
O: R1(X) R2(X) W3(Z) W3(X) C3 W2(Y) W2(X) C2 R4(X) W4(Z) C4 W1(Y) C1

- a. Appliquer le protocole de verrouillage à deux phases sur O jusqu'à l'exécution de toutes les actions. Quelle est l'exécution finale obtenue ?

- **V2P 01.5 pts**

| Transaction | Action | Demande de verrous | Réponse |
|-------------|--------|--------------------|-----------------------------------|
| T1 | R1(X) | SLOCK(X) | OK |
| T2 | R2(X) | SLOCK(X) | OK |
| T3 | W3(Z) | XLOCK(Z) | OK |
| T3 | W3(X) | XLOCK(X) | Non, T3 attend la fin de T1 et T2 |
| T2 | W2(Y) | XLOCK(Y) | OK |
| T2 | W2(X) | XLOCK(X) | Non, T2 attend la fin de T1 |
| T4 | R4(X) | SLOCK(X) | OK |
| T4 | W4(Z) | XLOCK(Z) | Non, T4 attend la fin de T3 |
| T1 | W1(Y) | XLCOCK(Y) | Non, T1 attend la fin de T2 |

- Graphe d'attente 0.5 pts



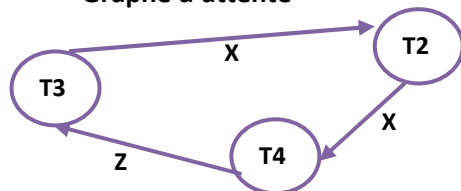
- De graphe d'attente, on constate qu'on a le cycle {T1, T2, T1} d'où on a un inter-blocage.

Pour lever l'inter blocage on choisit une transaction victime parmi {T1, T2} on l'annule et on la relance à la fin. La transaction T1 est la victime (n'a pas fait de mise à jour).

Rollback(T1): UNLOCK(X) et réveiller T2 0.5 pts

| | | | |
|----|-------|----------|-----------------------------|
| T2 | W2(X) | XLOCK(X) | Non, T2 attend la fin de T4 |
|----|-------|----------|-----------------------------|

- Graphe d'attente



- De graphe d'attente, on constate qu'on a le cycle {T2, T4, T3, T2} d'où on a un deuxième inter-blocage.

Pour lever l'inter blocage on choisit une transaction victime parmi {T2, T3, T4} on l'annule et on la relance à la fin. La transaction T4 est la victime (n'a pas fait de mise à jour).

Rollback(T4): UNLOCK(X) et réveiller T2 0.5 pts

| | | | |
|----|-------|----------|---|
| T2 | W2(X) | XLOCK(X) | OK; Fin de T2 ; Unlock(X), Unlock(Y) ; Réveiller T3 |
| T3 | W3(X) | XLOCK(X) | OK ; Fin de T3 ; Unlock(X), Unlock(Z) ; |

Relancer T1 dès son début :

| | | | |
|----|-------|-----------|--|
| T1 | R1(X) | SLOCK(X) | OK |
| T1 | W1(Y) | XLCOCK(Y) | OK ; Fin de T1; Unlock(X), Unlock(Y) ; |

Relancer T4 dès son début :

| | | | |
|----|-------|----------|---|
| T4 | R4(X) | SLOCK(X) | OK |
| T4 | W4(Z) | XLOCK(Z) | OK ; Fin de T4 ; Unlock(X), Unlock(Z) ; |

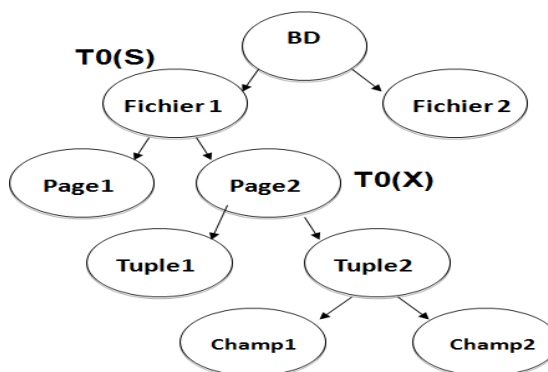
L'exécution série équivalente: **T2 T3 T1 T4** après le levé d'inter-blocage.

- b. Si la stratégie wound-wait est utilisée pour traiter les demandes de verrouillage, dans quel ordre les transactions sont finalement validées? **01.5 pts**

Rappel : stratégie WoundWait : seule une transaction plus récente peut attendre une transaction plus ancienne, dans le cas contraire la transaction récente, est annulée et reprise.

| Transaction | Action | Demande de verrous | Réponse |
|---|--------|--------------------|--|
| T1 | R1(X) | SLOCK(X) | OK |
| T2 | R2(X) | SLOCK(X) | Non, T2 attend T1 |
| T3 | W3(Z) | XLOCK(Z) | OK |
| T3 | W3(X) | XLOCK(X) | Non, T3 attend T1 |
| T2 | W2(Y) | XLOCK(Y) | T2 est en attente |
| T2 | W2(X) | XLOCK(X) | T2 est en attente |
| T4 | R4(X) | SLOCK(X) | Non, T4 attend T1 |
| T4 | W4(Z) | XLOCK(Z) | T4 est en attente |
| T1 | W1(Y) | XLCOCK(Y) | OK; Fin de T1; Unlock(X), Unlock(Y) ; Réveiller T2 |
| T2 | R2(X) | SLOCK(X) | OK |
| T2 | W2(Y) | XLOCK(Y) | OK |
| T2 | W2(X) | XLOCK(X) | OK; Unlock(X), Unlock(Y) ; Réveiller T3 |
| T3 | W3(X) | XLOCK(X) | OK ; Fin de T3 ; Unlock(X), Unlock(Z) ; Réveiller T4 |
| T4 | R4(X) | SLOCK(X) | OK |
| T4 | W4(Z) | XLOCK(Z) | OK ; Fin de T4 ; Unlock(X), Unlock(Z) ; |
| L'exécution série équivalente: T1 T2 T3 T4 . | | | |

3. Considérons une base de données contenant les enregistrements présentés ci-dessous et qui met en œuvre le verrouillage hiérarchique : **S** pour un verrou de partage et **X** verrou exclusif.



Supposons que cette base de données n'exécute actuellement qu'une seule transaction T_0 . Cette transaction a déjà acquis les verrous dont elle a besoin, comme le montre la figure. Pour chacune des autres transactions ci-dessous, indiquer si elle peut acquérir les verrous dont elle a besoin et s'exécuter. Si oui, indiquer les verrous qu'elle acquiert ; sinon, expliquer la raison.

- a. T_1 : Lecture de Page 1. **0.5 pts**

Oui, T_1 obtient le verrou S sur la page 1 car Page 1 est verrouillée en partage ($T_0(S)$) sur fichier 1

et ses descendants.

- b. T2 : Insérer Page 3 sous Fichier 1. 0.5 pts

Non, Fichier 1 est verrouillé en partage par T0.

- c. T3 : Suppression de Tuple 1. 0.5 pts

Non, tuple 1 est verrouillé en exclusif par T0.

4. Soient quatre transactions T1, T2, T3 et T4. Nous supposons le contenu suivant du journal de transaction :

| N° de la ligne dans le log | Instruction dans le log |
|----------------------------|-------------------------|
| 1 | <START T1> |
| 2 | <T1, A, 45, 10> |
| 3 | <START T2> |
| 4 | <T2, B, 5, 15> |
| 5 | <T2, C, 35, 10> |
| 6 | <T1, D, 15, 5> |
| 7 | <COMMIT T1> |
| 8 | <START T3> |
| 9 | <T3, A, 10, 15> |
| 10 | <CHECKPOINT1> |
| 11 | <T2, D, 5, 20> |
| 12 | <COMMIT T2> |
| 13 | <START T4> |
| 14 | <T4, D, 20, 30> |
| 15 | <T3, C, 10, 15> |
| 16 | <COMMIT T3> |
| 17 | <CHECKPOINT 2> |
| 18 | <COMMIT T4> |

Quelles sont les valeurs des granules A, B, C et D après la récupération :

- a. Si le système tombe en panne juste avant que la ligne 10 soit écrite sur le disque? 0.5 pts

Etats des transactions avant la panne

T1 est partialement validée

T2 et T3 sont actives.

Les 3 transactions non rien écrits sur la BD le CHECKPOINT1 n'est pas encore arrivé.

T2 et T3 rien à faire.

Refaire les mises à jour de T1 d'où l'état suivant des granules :

(A, B, C, D) = (10, 5, 35, 5)

- b. Si le système tombe en panne juste avant que la ligne 13 soit écrite sur le disque? 0.5 pts

Etats des transactions avant la panne

T1 est validée terminée.

T2 est partialement validée.

T3 est active.

T1 rien à faire.

T2 refaire les mises à jour.

T3 défaire les mises à jour effectuées en CHECKPOINT1

d'où l'état suivant des granules :

(A, B, C, D) = (10, 15, 10, 20)

c. Si le système tombe en panne juste avant que la ligne 18 soit écrite sur le disque ? **0.5 pts**

Etats des transactions avant la panne

T1 et T2 sont validées terminées.

T3 est partialement validée.

T4 est active.

T1 et T2 rien à faire.

T3 refaire les mises à jour.

T4 défaire les mises à jour effectuées en CHECKPOINT2

d'où l'état suivant des granules :

(A, B, C, D) = (15, 15, 15, 20)

Bon courage