



INF3710 –Fichiers et Bases de données

Hiver 2019

TP No. 4

Groupe 2

1852132 – Kenny Lui

1893058 – Gabriel Houle-Violette

Soumis à: Julien Bergeron

24 mars 2019

Partie 2

- 1) $\sigma_{annee = 2010}(\text{Spectacle})$
- 2) $\sigma_{age < 20 \wedge age > 29}(\text{Danseur})$
- 3) $\pi_{nom}(\sigma_{nationalite='canadien'}(\text{DirecteurArtistique}))$
- 4) $\pi_{Danseur.nom, Spectacle.titre}(\text{Performance} \bowtie \text{Danseur} \bowtie \text{Spectacle})$
- 5) $perf_cygne \leftarrow \sigma_{role='cygne'}(\text{Performance})$
 $\pi_{Danseur.nom, Spectacle.annee}(perf_cygne \bowtie \text{Danseur} \bowtie \text{Spectacle})$
- 6) $spect_opus_cactus \leftarrow \sigma_{titre=Opus\ Cactus}(\text{Spectacle})$
 $\pi_{Danseur}(spect_opus_cactus \bowtie \text{Danseur} \bowtie \text{Spectacle})$
- 7) $spect_phil \leftarrow \pi_{Spectacle.titre}(\text{Performance} \bowtie \text{Danseur} \bowtie \text{Spectacle})$
 $spect_kate \leftarrow \pi_{Spectacle.titre}(\text{Performance} \bowtie \text{Danseur} \bowtie \text{Spectacle})$
 $spect_phil \cap spect_kate$

8)

a.r. :

$\rho_R(\text{AgeMoyen}) \ni \text{AVG age}(\text{Danseur})$

Sql :

Select AVG(age) as AgeMoyen FROM Danseur;

9)

a.r. :

$lucie_id \leftarrow \pi_{id}(\sigma_{nom='Lucie\ tremblay'}(\text{Danseur}))$

$spect_w_lucie \leftarrow \pi_{Spectacle}(\text{Performance} \bowtie lucie_id \bowtie \text{Spectacle})$

$spect_wo_lucie \leftarrow \text{Spectacle} \text{ --- } spect_w_lucie$

$\pi_{Danseur.nom}(\text{Performance} \bowtie spect_wo_lucie \bowtie \text{Danseur})$

Sql

(Select d.nom

From Danseur d, Performance P

WHERE d.danseurID = P.danseurID AND P.SpectacleId NOT IN)

(Select g.SpectacleId

FROM Danseur g, Performance P

WHERE g.DanseurId = Performance.DanseurId AND g.nom = 'Lucie Tremblay')

10)

a.r.

$\rho_R(\text{nbSpectacle}) \preceq \text{Count}(\sigma_{\text{DanseurId}=1}(\text{Performance}))$

sql

Select COUNT(*) as nbSpectable

From Performance p

Where p.DanseurId = 1;

11)

a.r.

$\text{DanseursId} \leftarrow \pi_{\text{DanseurId}}(\text{Danseur})$

$\text{SpectId} \leftarrow \pi_{\text{SpectacleId}}(\text{Spectacle})$

$\text{Spect} \leftarrow \pi_{\text{Spectacle}}(\text{Performance} \bowtie \text{DanseurId} \bowtie \text{SpectId})$

Sql:

(Select d

From Danseur d, Performance

LEFT JOIN Performance.SpectacleId

ON d.DanseurId = Performance.SpectacleId;

12)

a.r. :

$\rho_R(\text{nbCatégories}) \preceq \text{Count}(\pi_{\text{Spectacle.categorie}}(\text{Spectacle}))$

sql :

Select COUNT(*) as nbCatégorie

From Spectacle s;

13)

a.r. :

$\text{DanseursId} \leftarrow \pi_{\text{id}}(\text{Danseur})$

$\text{spect_w_danseurs} \leftarrow \pi_{\text{Spectacle}}(\text{Performance} \bowtie \text{DanseursId} \bowtie \text{Spectacle})$

$\text{spect_wo_danseurs} \leftarrow \text{Spectacle} \text{ --- } \text{spect_w_danseurs}$

$\pi_{\text{Danseur}}(\text{Performance} \bowtie \text{spect_wo_danseurs} \bowtie \text{Danseur})$

Sql

(Select d

From Danseur d, Performance P

WHERE d.danseurID = P.danseurID AND P.SpectacleId NOT IN)

(Select g.SpectacleId

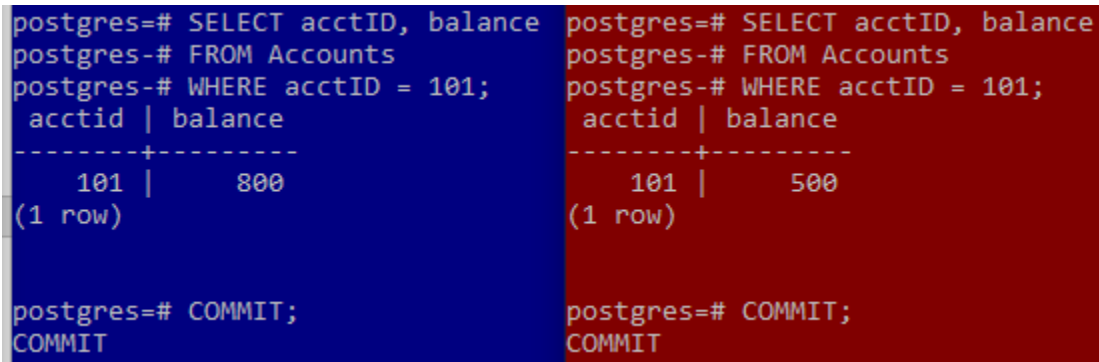
FROM Danseur g, Performance P

WHERE g.DanseurId = Performance.DanseurId)

Partie 3 : Transactions

1)

a)



```
postgres=# SELECT acctID, balance
postgres=# FROM Accounts
postgres=# WHERE acctID = 101;
 acctid | balance
-----+-----
    101 |      800
(1 row)

postgres=# COMMIT;
COMMIT
```

```
postgres=# SELECT acctID, balance
postgres=# FROM Accounts
postgres=# WHERE acctID = 101;
 acctid | balance
-----+-----
    101 |      500
(1 row)

postgres=# COMMIT;
COMMIT
```

Le problème est qu'une donnée est modifiée en concurrence, mais l'instruction de modification de la session A est commit alors que la session B a accédé à la vieille version de la donnée pour y effectuer sa modification. Ainsi, la modification de la session B efface celle de la session A et on perd l'information de mise à jour de A.

b)

Il s'agit de verrouiller le tuple dont l'acctID est 101 à l'aide de SELECT FOR UPDATE pour chaque transaction :

```
SELECT balance - 200 as bal into balancea
-- on rajoute FOR UPDATE pour verrouiller le tuple
FROM Accounts WHERE acctID = 101 FOR UPDATE;

SELECT balance - 500 as bal into balanceb
-- on rajoute FOR UPDATE pour verrouiller le tuple
FROM Accounts WHERE acctID = 101 FOR UPDATE;
```

2)

a)

Le niveau d'isolation étant à READ COMMITTED, la session A ne verra pas les modifications apportées par la session B avant le COMMIT de celle-ci. Cela pourrait porter problème si les deux transactions sont exécutées en même temps.

b)

Il n'y aura aucune différence puisque REPEATABLE READ ne voit que les modifications qui sont apportées dans la même transaction et puisque ces modifications sont faites dans la transaction B, le problème trouvé en a) reste inchangé.

c)

Le niveau d'isolation de la transaction A est en REPEATABLE READ et ne voit donc pas les modifications qui n'ont pas été COMMIT par les autres transactions. Ainsi, toutes les modifications apportées par la transaction B sont ignorées puisqu'elles ne sont jamais COMMIT.