Eric THIERRY M2CCI

**TP2 Base de données**

**Agence de voyage**

ATTENTION VERIFIER SOUS-REQUETES POUR R13

*14/12/16*

**Requête 9 : Donner la liste des numéros des circuits qui passent dans toutes les villes d’un pays donné.**

Schéma et spécification :

R9 (numC)<x> ∈ R9 ⟺ Le circuit de numéro x passe par toutes les villes du pays donné.

Requête SQL :

ACCEPT nomPays PROMPT 'Donner un nom de pays (ex : France) : ';

SELECT numC

FROM (SELECT numC, COUNT(ville) AS villeVisitee

FROM (SELECT nomV AS ville

FROM AGENCE.LesVilles

WHERE pays =&nomPays)A

/\*A(ville)<x> ∈ A  La ville x est une ville du pays donné\*/

NATURAL JOIN (SELECT numC, vDep AS ville FROM AGENCE.LesCircuits

union

SELECT numC, vArr

FROM AGENCE.LesCircuits

union

SELECT numC, vEtape

FROM AGENCE.LesEtapes)B

/\*B(numC, ville)<x,y> ∈ B ⟺ La ville y fait partie du circuit x\*/

GROUP BY numC)C

/\*C(numC, villeVisitee)<x,y> ∈ C  Le circuit de numéro x visite y ville du pays donné\*/

JOIN (SELECT COUNT(nomV) AS nbville

FROM AGENCE.LesVilles

WHERE pays = &nomPays)D

/\*D(nbVille)<x> ∈ à D ⟺ x est le nombre de ville du pays donné\*/

on (nbville = villeVisitee);

Résultats et tests de la requête SQL et des sous-requêtes :

***Résultat de la requête R9 avec Norvège comme pays donné :***

R9 (numC)<x> ∈ R9 ⟺ Le circuit de numéro x passe par toutes les villes du pays donné.

Donner un nom de pays (ex : France) : Norvege

NUMC

----------

16

21

***Tests de la requête R9 :***

Pour les tests on suppose que le que l’utilisateur rentre un pays P.

1. Vérifier que le circuit numéro X passe par toutes les villes du pays P. Le numéro X doit sortir dans les résultats de la requête.

Donner un nom de pays (ex : France) : Norvege

NUMC

----------

16

21

Le test est passant car les circuits 16 et 21 passent bien par toutes les villes de Norvège.

1. Vérifier que le circuit X passe par toutes les villes du pays P mais aussi par des villes d’un ou plusieurs autres pays. Le numéro X doit sortir dans les résultats de la requête.

Donner un nom de pays (ex : France) : Norvege

NUMC

----------

16

21

Le test est passant car le circuit 16 passe par une ville de Finlande et toutes les villes de Norvège. Et le circuit 21 passe par une ville de Finlande, toutes les villes de Norvège et quelques villes d’Italie.

1. Vérifier que le circuit X passe par certaines villes du pays P. Le numéro X ne doit pas sortir dans les résultats de la requête.

Donner un nom de pays (ex : France) : Italie

no rows selected

Le test est passant, le circuit 21 vu précédemment passe par une partie des villes d’Italie et ne sort donc pas dans les résultats.

1. Vérifier que le circuit X a pour ville de départ ou d’arrivée une ville du pays P et passe par toutes les autres villes du pays P. Le numéro X doit sortir dans les résultats de la requête.

Ce test est sortant car les tables de départ ne permettent pas de le vérifier. Afin qu’il soit passant on doit ajouter les villes d’Islande manquantes au circuit numéro 14. Et ainsi ce numéro de circuit figurerait dans les résultats.

Modification de la table lesEtapes, les données ajoutées sont soulignées :

insert into LesETAPEs values (14,1,'Reykjavik',3);

insert into LesETAPEs values (14,2,'Gullfoss',2);

insert into LesETAPEs values (14,3,'Geysir',2);

insert into LesETAPEs values (14,4,'Hveravellir',2);

insert into LesETAPEs values (14,5,'Akureyri',2);

insert into LesETAPEs values (14,6,'Myvatn',2);

insert into LesETAPEs values (14,7,'Asbyrgi',2);

insert into LesETAPEs values (14,8,'Godafoss',2);

insert into LesETAPEs values (14,9,'Husavik',2);

insert into LesETAPEs values (14,10,'Myvatn',2);

insert into LesETAPEs values (14,11,'Reykholt',2);

NUMC

----------

14

Le test est maintenant passant. Le circuit 14 apparait bien dans les résultats avec Islande comme pays donné.

***Résultat de la sous-requête A avec Norvège comme pays donné :***

A(ville)<x> ∈ A ⟺ La ville x est une ville du pays donné

VILLE

-------------

Cap Nord

Hammerfest

Inari

Saarijarvi

Salla

Tornio

Ylivieska

***Tests de la sous-requête A :***

1. Vérifier que toutes les villes du pays P sortent dans les résultats de la requête.

Donner un nom de pays (ex : France) : Norvege

VILLE

-------------

Cap Nord

Hammerfest

Inari

Saarijarvi

Salla

Tornio

Ylivieska

Le test est passant toutes les villes ne Norvège sont présente dans les résultats.

1. Vérifier que l’ajout ou la suppression d’une ville du pays P, modifie bien la liste des villes sortant pour le pays P.

Après ajout de la ville d’Oslo dans la table les villes, elle apparait dans les résultats de la requête.

insert into LesVILLEs values ('Oslo','Norvege');

VILLE

--------------------

Oslo

Cap Nord

Hammerfest

Inari

Saarijarvi

Salla

Tornio

Ylivieska

Après suppression de la ville du Cap Nord de la table les villes, elle ne figure plus dans les résultats de la requête.

~~insert into LesVILLEs values ('Cap Nord','Norvege');~~

VILLE

--------------------

Hammerfest

Inari

Saarijarvi

Salla

Tornio

Ylivieska

***Résultat de la sous-requête B:***

B(numC, ville)<x,y> ∈ B ⟺ La ville y fait partie du circuit x

NUMC VILLE

---------- -------------

1 Londres

1 Paris

2 Londres

2 Paris

3 Amsterdam

3 Paris

4 Amsterdam

4 Paris

-------------

-------------

-------------

20 Briancon

21 Cap Nord

21 Florence

21 Hammerfest

21 Helsinky

21 Inari

21 Paris

21 Ravenne

21 Rome

21 Saarijarvi

21 Salla

21 Tornio

21 Venise

21 Verone

21 Ylivieska

123 rows selected.

1. Vérifier que toutes les villes du circuit X sortent dans les résultats de la requête.

NUMC VILLE

---------- -------------

17 Castelo de Vide

17 Evora

17 Faro

17 Fatima

17 Leiria

17 Lisbonne

17 Porto

17 Urgeirica

17 Vila Real

Le test est passant toutes les villes du circuit 17 sont présente dans les résultats.

1. Vérifier que l’ajout ou la suppression d’une ville du circuit numéro X, modifie bien la liste des villes sortant pour le circuit numéro X.

Ajout de la ville Lyon, au circuit 17 : fait bien apparaitre la ville de Lyon dans les résultats de la requête.

insert into LesETAPEs values (17,1,'Lyon',2);

NUMC VILLE

---------- -------------

17 Lyon

17 Castelo de Vide

17 Evora

17 Faro

17 Fatima

17 Leiria

17 Lisbonne

17 Porto

17 Urgeirica

17 Vila Real

Suppression de la ville Evora, au circuit 17 : donnerait le même résultat que la requête en 1) aves l’absence d’Evora

~~insert into LesETAPEs values (17,2,'Evora',2);~~

NUMC VILLE

---------- -------------

17 Castelo de Vide

17 Faro

17 Fatima

17 Leiria

17 Lisbonne

17 Porto

17 Urgeirica

17 Vila Real

***Résultat de la sous-requête C avec Norvège comme pays donné :***

C(numC, villeVisitee)<x,y> ∈ C ⟺ Le circuit de numéro x visite y ville du pays donné

NUMC VILLEVISITEE

---------- ------------

16 7

21 7

***Tests de la sous-requête C :***

1. Vérifier que le circuit X visite V villes du pays P. Sort dans la table les circuits X avec V villes correspondantes au pays P.

Donner un nom de pays (ex : France) : Italie

NUMC VILLEVISITEE

---------- ------------

6 1

5 1

18 5

19 13

21 5

Le test est passant, un certain nombre de circuit passe par une ou plusieurs villes italiennes.

1. Vérifier que le circuit X n’ayant pas de ville visités pour le pays P ne sorte pas dans les résultats de la requête.

Donner un nom de pays (ex : France) : Italie

NUMC VILLEVISITEE

---------- ------------

6 1

5 1

18 5

19 13

21 5

Le test est passant, le circuit 1 ne passe par aucune ville italienne et donc il ne figure pas dans les résultats.

1. Vérifier que l’ajout ou la suppression d’une ville du pays P dans le circuit X modifie ne nombre de ville visité.

Ajouter la ville italienne ‘Vérone’ au circuit 6

insert into LesETAPEs values (6,1,'Verone',2);

Donner un nom de pays (ex : France) : Italie

NUMC VILLEVISITEE

---------- ------------

6 2

5 1

18 5

19 13

21 5

Supprimer la ville italienne ‘Vérone’ au circuit 18

~~insert into LesETAPEs values (18,1,'Verone',2);~~

Donner un nom de pays (ex : France) : Italie

NUMC VILLEVISITEE

---------- ------------

6 2

5 1

18 4

19 13

21 5

**Requête 11 : Donner le numéro, le prix de base (sans tenir compte du prix des monuments visités), la date de départ et le nombre de places disponibles et dont le nombre de jours est inférieur ou égale à un entier donné.**

Schéma et spécification de la requête :

R11(numC, dateDep, prix, Dispo)<w,x,y,z> ∈ R11 ⟺ le circuit de numéro *w* a pour date de départ *y*. Son prix est *x* et il reste *z* places disponibles.

Requête SQL :

ACCEPT choixJour PROMPT 'Choisissez le nombre de jours pour votre voyage :';

SELECT numC, dateDep, prix, Dispo

FROM (SELECT numC, dateDep, prix, (nbplaces - nvl(nbReserve, 0)) AS Dispo

FROM AGENCE.LesProgrammations NATURAL LEFT OUTER JOIN

(SELECT numC, datedep, SUM(nbRes) AS nbReserve FROM AGENCE.LesReservations

GROUP BY numC, datedep)A

/\*A(numC, dateDep, nbReserve)<x,y,z> ∈ A ⟺ le circuit de numéro *x* a pour date de départ *y* et possède *z* places de réservées\*/

NATURAL JOIN AGENCE.LesCircuits

WHERE (nbplaces - nvl(nbReserve, 0)) > 0)B

/\*B(numC, dateDep, prix, Dispo)<w,x,y,z> ∈ B ⟺ le circuit de numéro *w* a pour date de départ *x*. Son prix est *y* et il reste *z* places disponibles. \*/

NATURAL JOIN (SELECT numC, SUM(nbjours) AS totaljour FROM AGENCE.LesEtapes

GROUP BY numC

HAVING SUM(nbjours) <= &choixJour)C

/\*C(numC, totaljour)<y,z> ∈ C ⟺ Le circuit de numéro *y* dure *z* jours \*/

ORDER BY numC, dateDep;

Résultats et test de la requête SQL et des sous-requêtes :

***Résultat de la requête R11 avec 4 comme entier entré :***

R11(numC, dateDep, prix, Dispo)<w,x,y,z> ∈ R11 ⟺ le circuit de numéro *w* a pour date de départ *y*. Son prix est *x* et il reste *z* places disponibles.

Choisissez le nombre de jours pour votre voyage : 4

NUMC DATEDEP PRIX DISPO

---------- --------- ---------- ----------

1 04-JAN-10 1160 34

1 04-FEB-10 1160 8

1 06-FEB-10 1160 34

1 24-JUL-10 1160 10

2 06-JAN-10 1160 2

2 07-JAN-10 1160 9

2 06-FEB-10 1160 12

2 05-SEP-10 1160 30

3 24-DEC-09 1040 13

3 31-DEC-09 1040 45

3 03-JUL-10 1040 1

4 30-JUN-10 1270 11

4 06-AUG-10 1270 10

4 31-AUG-10 1270 1

4 06-NOV-10 1270 18

6 06-FEB-10 2520 22

6 06-SEP-10 2520 10

6 06-OCT-10 2520 12

6 16-NOV-10 2520 34

***Test de la requête R11 :***

On note E l’entier entré par l’utilisateur

1. Vérifier que le circuit numéro X a encore des places disponibles et a une durée inférieure ou égale à E. Le circuit numéro X doit sortir dans les résultats la requête.

Choisissez le nombre de jours pour votre voyage : 4

NUMC DATEDEP PRIX DISPO TOTALJOUR

---------- --------- ---------- ---------- ----------

1 04-JAN-10 1160 34 2

1 04-FEB-10 1160 8 2

Le test est passant, exemple avec le circuit 1 pour les programmations du 04-JAN-10 et 04-FEB-10

1. Vérifier que le circuit X a encore des places disponibles et a une durée strictement supérieure à E. Le circuit numéro X ne doit pas sortir dans les résultats de la requête.

NUMC DATEDEP PRIX DISPO TOTALJOUR

---------- --------- ---------- ---------- ----------

5 31-AUG-10 2740 64 5

5 06-NOV-10 2740 3 5

Ce test est passant, on voit que le circuit 5 a des places disponibles et se déroule sur 5 jours, et il est bien absent des résultats de la requête pour une durée maximum de 4 jours.

1. Vérifier que le circuit X n’a plus de places disponibles et a une durée inférieure ou égale à E. Le circuit numéro X ne doit pas sortir dans les résultats de la requête.

Le test est passant pour E = 4, la programmation du 21-JUL-10 pour le circuit 1 a une durée de 2 jours et n’a plus de places disponibles. Cette ligne est bien absente du résultat de la requête.

1. Vérifier que le circuit X n’a plus de places disponibles et a une durée strictement supérieure à E. Le circuit numéro X ne doit pas sortir dans les résultats de la requête.

Le test est passant pour E = 4, la programmation du 14-FEB-10 pour le circuit 8 a une durée de 7 jours et n’a plus de places disponibles. Cette ligne est bien absente du résultat de la requête.

***Résultat de la sous-requête A :***

A(numC, dateDep, nbReserve)<x,y,z> ∈ A ⟺ le circuit de numéro *x* a pour date de départ *y* et possède *z* places de réservées

NUMC DATEDEP NBRESERVE

---------- --------- ----------

8 28-FEB-10 7

3 03-JUL-10 11

7 16-DEC-09 51

13 01-JAN-10 10

12 06-FEB-10 79

18 06-DEC-10 11

8 14-FEB-10 1

8 16-FEB-10 1

4 30-JUN-10 88

5 31-AUG-10 2

2 07-JAN-10 25

19 15-APR-10 20

5 06-NOV-10 43

21 15-JAN-10 24

13 31-DEC-09 48

14 26-JUL-10 21

20 27-JAN-10 5

1 21-JUL-10 10

2 05-FEB-10 99

1 04-FEB-10 4

10 01-JAN-10 2

9 30-OCT-10 10

10 11-FEB-10 2

***Tests de la sous-requête A :***

1. Vérifier qu’un circuit X ayant au moins une place réservée pour une programmation P sort dans les résultats de la requête.

NUMC DATEDEP NBRESERVE

---------- --------- ----------

8 28-FEB-10 7

3 03-JUL-10 11

7 16-DEC-09 51

13 01-JAN-10 10

12 06-FEB-10 79

18 06-DEC-10 11

8 14-FEB-10 1

8 16-FEB-10 1

4 30-JUN-10 88

5 31-AUG-10 2

2 07-JAN-10 25

19 15-APR-10 20

5 06-NOV-10 43

21 15-JAN-10 24

13 31-DEC-09 48

14 26-JUL-10 21

20 27-JAN-10 5

1 21-JUL-10 10

2 05-FEB-10 99

1 04-FEB-10 4

10 01-JAN-10 2

9 30-OCT-10 10

10 11-FEB-10 2

Le test est passant

1. Vérifier qu’un circuit X ayant aucune réservation pour une programmations P ne figure pas dans les résultats de la requête.

Le test set passant, la programmation du 04-JAN-10 pour le circuit 1 ne sort pas car il n’y a aucune réservation.

***Résultat de la sous-requête B :***

B(numC, dateDep, prix, Dispo)<w,x,y,z> ∈ B ⟺ le circuit de numéro *w* a pour date de départ *x*. Son prix est *y* et il reste *z* places disponibles.

NUMC DATEDEP PRIX DISPO

---------- --------- ---------- ----------

1 24-JUL-10 1160 10

1 06-FEB-10 1160 34

1 04-FEB-10 1160 8

1 04-JAN-10 1160 34

2 05-SEP-10 1160 30

2 06-FEB-10 1160 12

2 07-JAN-10 1160 9

2 06-JAN-10 1160 2

3 03-JUL-10 1040 1

3 31-DEC-09 1040 45

3 24-DEC-09 1040 13

----------------------------------

----------------------------------

----------------------------------

19 20-DEC-10 10400 11

19 10-DEC-10 10400 28

19 16-SEP-10 10400 10

19 06-SEP-10 10400 20

19 05-AUG-10 10400 12

19 15-APR-10 10400 25

19 06-FEB-10 10400 10

21 06-APR-10 15000 99

21 26-FEB-10 15000 12

21 25-JAN-10 15000 3

21 15-JAN-10 15000 42

20 03-APR-10 450 87

20 01-APR-10 450 13

20 06-FEB-10 450 1

20 02-FEB-10 450 14

20 20-JAN-10 450 12

20 22-DEC-09 450 18

87 rows selected.

***Tests de la sous-requête B :***

1. Vérifier que le circuit X avec la programmation P possède encore des places disponibles. La programmation P du circuit X doit sortir dans les résultats de la requête.

NUMC DATEDEP PRIX DISPO

---------- --------- ---------- ----------

1 24-JUL-10 1160 10

1 06-FEB-10 1160 34

1 04-FEB-10 1160 8

1 04-JAN-10 1160 34

Le test est passant, les programmations pour le circuit 1 ayant encore des places disponibles sont présentes dans les résultats de la requête.

1. Vérifier que le circuit X avec la programmation P n’ayant plus de places disponibles ne sort pas dans les résultats de la requête.

Le test est passant. La programmation du 21-JUL-10 pour le circuit 1 n’a plus de places disponibles. Cette ligne est bien absente du résultat de la requête.

***Résultat de la sous-requête C :***

C(numC, totaljour)<y,z> ∈ C ⟺ Le circuit de numéro y dure z jours

Choisissez le nombre de jours pour votre voyage : 4

NUMC TOTALJOUR

---------- ----------

1 2

2 2

3 3

4 4

6 4

***Tests de la sous-requête C :***

1. Vérifier que le circuit X ayant une durée inférieure ou égale à l’entier E sort dans les résultats de la requête.

Choisissez le nombre de jours pour votre voyage : 4

NUMC TOTALJOUR

---------- ----------

1 2

2 2

3 3

4 4

6 4

Le test est passant, les circuits 1, 2, 3, 4, 6 ont une durée inférieur ou égale à 4.

1. Vérifier que le circuit X ayant une durée supérieure à l’entier E ne sort pas dans les résultats de la requête.

Ce test est passant pour E = 4, le circuit numéro 5 n’apparait pas dans les résultats car sa durée est de 5 jours.

**Requête 13 : Pour chaque programmation de circuit, retrouver les pays dans lequel passe le circuit et la date à laquelle le circuit arrive dans ce pays**

Schéma et spécification de la requête :

R13(numC, dateDep, pays, dateEntree)<w,x,y,z> ∈ R13 ⟺ la programmation de la date x du circuit w arrive dans le pays y à la date z.

Requête SQL :

WITH X AS (

SELECT numC, rang, pays, nbJours

FROM (

SELECT numC, rang, vEtape AS ville, nbJours

FROM AGENCE.LesEtapes

UNION

SELECT numC, 0 AS rang, vDep, 0 AS nbJours

FROM AGENCE.LesCircuits

UNION

SELECT numC, nvl((rmax+1),1) AS rang, vArr, 0 AS nbJours

FROM AGENCE.LesCircuits

NATURAL LEFT OUTER JOIN ( SELECT numC, max(rang) AS rmax

FROM AGENCE.LesEtapes

GROUP BY numC ) )

JOIN AGENCE.LesVilles ON ( ville=nomV )

)

/\* X(numC, rang, pays, nbJours)<x,y,z,j> ∈ X ⟺ Le numéro de circuit x possède une ville de rang y ayant pour pays z et cette ville est visitée pendant j jour.\*/

SELECT numC, dateDep, pays, (dateDep+nvl(nbJ,0)) AS dateEntree

FROM (

SELECT X1.numC, X1.rang, X1.pays

FROM ( X ) X1

JOIN ( X ) X2

ON ( X1.numC=X2.numC AND X1.pays!=X2.pays AND X2.rang=(X1.rang-1) )

UNION

SELECT numC, rang, pays

FROM X

WHERE rang=0 ) A

/\* A(numC, rang, pays)<x,y,z> ∈ A ⟺ Le pays z est différent du pays de la ville de rang y précédente pour le circuit numéro x.\*/

NATURAL LEFT OUTER JOIN (

SELECT Y1.numC, Y1.rang, sum(Y2.nbJours) AS nbJ

FROM ( X ) Y1

JOIN ( X ) Y2

ON ( Y1.numC=Y2.numC AND Y1.rang>Y2.rang )

GROUP BY Y1.numC, Y1.rang ) B

/\* B(numC, rang, nbJ)<x,y,z> ∈ B ⟺ le nombre de jour z est la somme des jours passés dans les villes de rang y précédentes pour les circuits de numéro x.\*/

NATURAL JOIN AGENCE.LesProgrammations

ORDER BY numC, dateDep, dateEntree;

Résultats et tests de la requête SQL et des sous-requêtes :

***Résultat de la requête R13 :***

R13(numC, dateDep, pays, dateEntree)<w,x,y,z> ∈ R13 ⟺ la programmation de la date x du circuit w arrive dans le pays y à la date z.

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

1 04-JAN-10 France 04-JAN-10

1 04-JAN-10 Angleterre 04-JAN-10

1 04-JAN-10 France 06-JAN-10

1 04-FEB-10 Angleterre 04-FEB-10

1 04-FEB-10 France 04-FEB-10

1 04-FEB-10 France 06-FEB-10

1 06-FEB-10 Angleterre 06-FEB-10

1 06-FEB-10 France 06-FEB-10

1 06-FEB-10 France 08-FEB-10

1 21-JUL-10 France 21-JUL-10

1 21-JUL-10 Angleterre 21-JUL-10

1 21-JUL-10 France 23-JUL-10

1 24-JUL-10 Angleterre 24-JUL-10

1 24-JUL-10 France 24-JUL-10

1 24-JUL-10 France 26-JUL-10

2 06-JAN-10 France 06-JAN-10

2 06-JAN-10 Angleterre 06-JAN-10

2 06-JAN-10 France 08-JAN-10

2 07-JAN-10 Angleterre 07-JAN-10

2 07-JAN-10 France 07-JAN-10

2 07-JAN-10 France 09-JAN-10

------------------------------------------

------------------------------------------

------------------------------------------

20 22-DEC-09 France 22-DEC-09

20 20-JAN-10 France 20-JAN-10

20 27-JAN-10 France 27-JAN-10

20 02-FEB-10 France 02-FEB-10

20 06-FEB-10 France 06-FEB-10

20 01-APR-10 France 01-APR-10

20 03-APR-10 France 03-APR-10

21 15-JAN-10 France 15-JAN-10

21 15-JAN-10 Finlande 15-JAN-10

21 15-JAN-10 Norvege 17-JAN-10

21 15-JAN-10 Italie 25-JAN-10

21 25-JAN-10 Finlande 25-JAN-10

21 25-JAN-10 France 25-JAN-10

21 25-JAN-10 Norvege 27-JAN-10

21 25-JAN-10 Italie 04-FEB-10

21 26-FEB-10 Finlande 26-FEB-10

21 26-FEB-10 France 26-FEB-10

21 26-FEB-10 Norvege 28-FEB-10

21 26-FEB-10 Italie 08-MAR-10

21 06-APR-10 France 06-APR-10

21 06-APR-10 Finlande 06-APR-10

21 06-APR-10 Norvege 08-APR-10

21 06-APR-10 Italie 16-APR-10

198 rows selected.

***Tests de la requête R13 :***

1. Je vérifie que le circuit X commence, se déroule et se finit dans un seul pays, sort bien dans les résultats de la requête.

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

7 31-OCT-10 France 31-OCT-10

8 24-DEC-09 France 24-DEC-09

14 06-JUL-10 Islande 06-JUL-10

Le test est passant. Les circuits 7, 8 et 14 se déroule dans un seul et même pays, ils apparaissent bien qu’une seule fois dans la table résultat avec les dates de départ et d’entrée identiques.

1. Je vérifie que le circuit X commence, se déroule et se finit dans des pays différents, sort bien dans les résultats de la requête.

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

21 06-APR-10 France 06-APR-10

21 06-APR-10 Finlande 06-APR-10

21 06-APR-10 Norvege 08-APR-10

21 06-APR-10 Italie 16-APR-10

Le test est passant. Le circuit 21 est présent 4 fois pour la programmation du 06-APR-10) car il entre dans 4 pays différents.

1. Je vérifie que le circuit X a sa dernière étape et sa ville d’arrivée dans le même pays et passe dans un pays différent au cours du séjour, sort bien dans les résultats de la requête.

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

21 06-APR-10 France 06-APR-10

21 06-APR-10 Finlande 06-APR-10

21 06-APR-10 Norvege 08-APR-10

21 06-APR-10 Italie 16-APR-10

Le test est passant. Le circuit 21 a pour dernière étape et pour ville d’arriver deux villes italiennes. On a bien le pays Italie qui apparait une seule fois.

1. Je vérifie que le circuit X passant plusieurs fois dans le même pays avec des dates d’entrée différentes sort dans les résultats de la requête.

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

1 04-JAN-10 France 04-JAN-10

1 04-JAN-10 Angleterre 04-JAN-10

1 04-JAN-10 France 06-JAN-10

13 31-DEC-09 France 31-DEC-09

13 31-DEC-09 Danemark 31-DEC-09

13 31-DEC-09 Groenland 04-JAN-10

13 31-DEC-09 France 18-JAN-10

Le test est passant. Les circuits 1 et 13, le pays France est présent deux fois car les circuits entre dans ce pays à deux fois dans ce pays à des dates différentes.

1. Je vérifie qu’un circuit n’ayant pas de ville étape et des villes de départ et d’arrivée avec des pays différents sort dans la table des résultats.

Le test est sortant car aucune donnée ne permet de le vérifier.

Insertion des données suivantes dans la table pour vérifier le test :

insert into LesCIRCUITs values (22,'Paris','Londres',450);

insert into LesPROGRAMMATIONs values (22,to\_date ('01-02-2017','DD-MM-YYYY'),03);

insert into LesPROGRAMMATIONs values (22,to\_date ('16-10-2016','DD-MM-YYYY'),03);

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

22 16-OCT-16 France 16-OCT-16

22 16-OCT-16 Angleterre 16-OCT-16

22 01-FEB-17 Angleterre 01-FEB-17

22 01-FEB-17 France 01-FEB-17

23 06-MAR-11 France 06-MAR-11

Le test est maintenant passant, le circuit 22 ajouté aux données figure bien dans les résultats de la requête.

1. Je vérifie qu’un circuit n’ayant pas de ville étape et des villes de départ et d’arrivée avec un pays identique sort dans la table des résultats.

Le test est sortant car aucune donnée ne permet de le vérifier.

Insertion des données suivantes dans la table pour vérifier le test :

insert into LesCIRCUITs values (23,'Paris','Lyon',450);

insert into LesPROGRAMMATIONs values (23,to\_date ('06-03-2011','DD-MM-YYYY'),03);

NUMC DATEDEP PAYS DATEENTRE

---------- --------- -------------------- ---------

23 06-MAR-11 France 06-MAR-11

Le test est maintenant passant, le circuit 23 ajouté aux données figure bien dans les résultats de la requête.

***Résultat de la sous-requête X :***

X(numC, rang, pays, nbJours)<x,y,z,j> ∈ X ⟺ Le numéro de circuit x possède une ville de rang y ayant pour pays z et cette ville est visitée pendant j jour.

NUMC RANG PAYS NBJOURS

---------- ---------- -------------------- ----------

1 0 France 0

1 1 Angleterre 2

1 2 France 0

2 0 France 0

2 1 Angleterre 2

2 2 France 0

3 0 France 0

3 1 Hollande 3

3 2 France 0

---------------------------------------------

---------------------------------------------

---------------------------------------------

20 0 France 0

20 1 France 6

20 2 France 0

21 0 France 0

21 1 Finlande 2

21 2 Norvege 1

21 3 Norvege 1

21 4 Norvege 1

21 5 Norvege 1

21 6 Norvege 1

21 7 Norvege 1

21 8 Norvege 2

21 9 Italie 3

21 10 Italie 3

21 11 Italie 2

21 12 Italie 1

21 13 Italie 3

21 14 Italie 0

151 rows selected.

***Tests de la sous-requête X :***

1. Vérifier que le circuit X a bien un rang 0 pour le pays de départ et que le rang du pays d’arrivé est bien le plus grand.

NUMC RANG PAYS

---------- ---------- --------------------

1 0 France

1 1 Angleterre

1 2 France

2 0 France

2 1 Angleterre

2 2 France

Le test est passant. Exemple pour les circuits 1 et 2, on voit que la France étant le pays de départ et d’arrivé de ces circuits a bien les rang 0 et maximum.

***Résultat de la sous-requête A :***

A(numC, rang, pays)<x,y,z> ∈ A ⟺ Le pays z est différent du pays de la ville de rang y précédente pour le circuit numéro x.

NUMC RANG PAYS

---------- ---------- --------------------

1 0 France

1 1 Angleterre

1 2 France

2 0 France

2 1 Angleterre

2 2 France

3 0 France

3 1 Hollande

3 2 France

4 0 France

4 1 Hollande

4 2 France

5 0 France

5 1 Italie

5 2 France

6 0 France

6 1 Italie

6 2 France

7 0 France

8 0 France

9 0 France

9 1 Angleterre

9 7 France

10 0 Irlande

11 0 Irlande

12 0 Irlande

13 0 France

13 1 Danemark

13 2 Groenland

13 9 France

14 0 Islande

15 0 France

15 1 Islande

15 5 France

16 0 France

16 1 Finlande

16 2 Norvege

16 9 France

17 0 Portugal

18 0 France

18 1 Italie

18 6 France

19 0 Italie

20 0 France

21 0 France

21 1 Finlande

21 2 Norvege

21 9 Italie

***Tests de la sous-requête A :***

1. Vérifier que le circuit X a un pays de rang r différent du pays de rang r-1, sort dans les résultats de la requête.

NUMC RANG PAYS

---------- ---------- --------------------

13 0 France

13 1 Danemark

13 2 Groenland

13 9 France

21 0 France

21 1 Finlande

21 2 Norvege

21 9 Italie

Le test est passant, les circuits 13 et 21 ont bien des rangs croissants entre deux pays différents.

1. Vérifier que le circuit X n’a jamais deux pays identiques pour des rangs qui se suivent.

NUMC RANG PAYS

---------- ---------- --------------------

13 0 France

13 1 Danemark

13 2 Groenland

13 9 France

Le test est passant, le circuit 13 entre deux fois en France, mais il y a bien des pays de rangs différents entre les deux entrées en France

***Résultat de la sous-requête B :***

B(numC, rang, nbJ)<x,y,z> ∈ B ⟺ le nombre de jour z est la somme des jours passés dans les villes de rang y précédentes pour les circuits de numéro x.

NUMC RANG NBJ

---------- ---------- ----------

1 1 0

1 2 2

2 1 0

2 2 2

3 1 0

3 2 3

4 1 0

4 2 4

5 1 0

5 2 5

------------------------

------------------------

------------------------

20 1 0

20 2 6

21 1 0

21 2 2

21 3 3

21 4 4

21 5 5

21 6 6

21 7 7

21 8 8

21 9 10

21 10 13

21 11 16

21 12 18

21 13 19

21 14 22

130 rows selected.

***Tests de la sous-requête B :***

1. Vérifier que pour le circuit X le rang 1 sort avec un nombre de jour égale à 0

NUMC RANG NBJ

---------- ---------- ----------

1 1 0

1 2 2

2 1 0

2 2 2

3 1 0

3 2 3

Le test est passant. Les circuits 1, 2 et 3 ont bien un nombre de jours égale à 0 pour le rang 1.

1. Vérifier que le circuit X a bien pour chaque rang un nombre de jour égale à la somme des jours des rangs précédents.

NUMC RANG NBJ NBJOURS

---------- ---------- ---------- ------------

21 1 0 2

21 2 2 1

21 3 3 1

21 4 4 1

21 5 5 1

21 6 6 1

21 7 7 1

21 8 8 1

21 9 10 2

21 10 13 3

21 11 16 3

21 12 18 2

21 13 19 1

21 14 22 3

Le test est passant. Pour le circuit 21, chaque valeur de l’attribut NBJ correspond à la somme des lignes de NBJOURS de rangs précédents