BD : Gestion d'un zoo, TP1

Frances Tom, Gourdon Jérémie January 25, 2022

1 Complétion du schéma

Le code de création des tables complété est donné en annexe.

Nous avons de plus rajouté 2 tuples dans les tables LesCages et LesAnimaux pour avoir 2 cages contenant des léopards :

```
01 | insert into LesCages values (13 , 'fauve' , 2);
02 | insert into LesAnimaux values ('Magalie', 'femelle' , 'leopard', 'fauve', 'Italie',
2014, 13);
```

2 Requêtes SQL

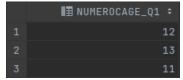
2.1 Dans quelles cages se trouvent les lions et les léopards?

Algèbre : (LesAnimaux : type_an = 'lion' OR type_an = 'leopard) [noCage]

Code:

```
01 | SELECT DISTINCT noCage AS numeroCage_Q1
02 | FROM LesAnimaux
03 | WHERE type_an='lion' OR type_an='leopard';
```

Résultat :



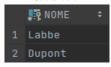
2.2 Quels sont les gardiens qui gardent des cages de léopards?

Algèbre : ((LesGardiens * LesAnimaux) : type_an = 'leopard') [nomE]

Code:

```
01 | SELECT DISTINCT nomE
02 | FROM LesGardiens NATURAL JOIN LesAnimaux
03 | WHERE type_an='leopard';
```

Résultat :



2.3 Quels sont les gardiens qui gardent à la fois des cages de léopards et de lions ?

Algèbre : ((LesGardiens * LesAnimaux) : type_an = 'leopard') [nomE] INTERSECT ((LesGardiens * LesAnimaux) : type_san = 'lion') [nomE]

Code:

```
01 | -- V1 --
02 | SELECT DISTINCT nomE
03 | FROM LesGardiens G JOIN LesAnimaux A1 USING (noCage)
```

```
04 | WHERE type_an='leopard'
05 l
     AND nomE IN ( SELECT nomE
06 I
                     FROM LesGardiens G JOIN LesAnimaux A1 USING (noCage)
                      WHERE type_an='lion');
07 |
08 I
     -- V2 --
09 |
     SELECT nomE
10 |
     FROM LesGardiens JOIN LesAnimaux USING (noCage)
11 l
     WHERE type_an='leopard'
     INTERSECT
13 l
14 |
     SELECT nomE
     FROM LesGardiens JOIN LesAnimaux USING (noCage)
15 |
     WHERE type_an='lion';
16 I
```

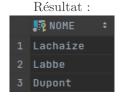
Résultat : NOME : Labbe

2.4 Quels sont les gardiens qui gardent des animaux qui sont ou ont été malades ?

 $\label{eq:Algèbre: ((LesMaladies * LesAnimaux) * LesGardiens) [nomE]} \\$

Code:

```
01 | SELECT DISTINCT nomE
02 | FROM LesMaladies JOIN LesAnimaux USING (nomA)
03 | JOIN LesGardiens USING (noCage);
```



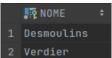
2.5 Quels sont les responsables qui ont ou ont eu des animaux malades dans leur allée ?

 $\label{eq:loss_equation} \mbox{Algèbre} : (((\mbox{LesResponsables * LesCages}) * \mbox{LesAnimaux }) * \mbox{LesMaladies }) \ [\mbox{nomE}]$

Code:

```
01 | SELECT DISTINCT nomE
02 | FROM LesResponsables JOIN LesCages USING (noAllee)
03 | JOIN LesAnimaux USING (noCage)
04 | JOIN LesMaladies USING (nomA);
```

Résultat :



2.6 Quels sont les gardiens qui ne gardent pas de cage de léopards?

Algèbre : (LesGardiens) [nomE] - ((LesGardiens * LesAnimaux) : type_an = 'leopard) [nomE]

Code:

```
01 | SELECT DISTINCT nomE
02 | FROM LesGardiens
03 | WHERE nomE NOT IN ( SELECT nomE
04 | FROM LesGardiens NATURAL JOIN LesAnimaux
05 | WHERE type_an='leopard');
```



2.7 Combien d'animaux sont malades?

Code:

```
01 | SELECT COUNT(DISTINCT nomA) AS NbAnimauxMalade
02 | FROM LesMaladies;
```



2.8 Combien y a-t-il d'animaux malades par type de maladie?

Code:

```
01 | SELECT nomM, COUNT(nomA) AS NbAnimauxMaladeParMaladie
02 | FROM LesMaladies
03 | GROUP BY nomM;
```



2.9 Quels sont les animaux avec au moins deux maladies?

Code:

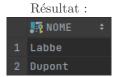
```
01 | SELECT nomA
02 | FROM LesMaladies
03 | GROUP BY nomA
04 | HAVING COUNT(nomM) >= 2;
```



2.10 Quels gardiens gardent plus de deux cages avec la fonction "fauve"?

Code:

```
01 | SELECT nomE
02 | FROM LesGardiens NATURAL JOIN LesCages
03 | WHERE fonction='fauve'
04 | GROUP BY nomE
05 | HAVING COUNT(noCage) >= 2;
```



2.11 Quels responsables possèdent le plus de cages dans une de ses allées ?

Pour cette requête, nous avons supposé que toutes les allées devaient avoir un responsbable. Code :

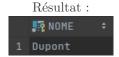
```
01 | SELECT nomE
02 | FROM LesResponsables NATURAL JOIN LesCages
03 | GROUP BY nomE, noAllee
04 | HAVING COUNT(noCage) = (SELECT MAX(COUNT(noCage))
05 | FROM LesCages
06 | GROUP BY noAllee);
```



2.12 Quels sont les gardiens qui gardent toutes les cages de léopards?

Pour cette requête, nous avons supposé que toutes les allées devaient avoir un responsbable. Code :

```
01 | SELECT nomE
02 | FROM LesGardiens NATURAL JOIN LesAnimaux
03 | WHERE type_an='leopard'
04 | GROUP BY nomE
05 | HAVING COUNT(DISTINCT noCage) = ( SELECT COUNT(DISTINCT noCage)
06 | FROM LesAnimaux
07 | WHERE type_an='leopard');
```



2.13 Pour chaque gardien, donner le nombre de léopards gardés ? (on veut tous les gardiens)

Code:

```
01 | SELECT nomE, NVL(COUNT(nomA), 0) AS NbLeopardsGardes
02 | FROM LesGardiens G LEFT JOIN LESANIMAUX A ON (A.noCage=G.noCage AND type_an='leopard')
03 | GROUP BY nomE;
```

Résultat :



2.14 Pour chaque gardien, donner le nombre de cage de léopards gardées ? (on veut tous les gardiens)

Code:

```
01 | SELECT nomE, nvl(COUNT(DISTINCT A.noCage), 0) AS NbCagesLeopardsGardees
02 | FROM LesGardiens G LEFT JOIN LesAnimaux A ON (A.noCage=G.noCage AND A.type_an='leopard')
03 | GROUP BY nomE;
```

Résultat :



3 Annexe

```
drop table LesMaladies ;
02 |
     drop table LesAnimaux ;
     drop table LesResponsables ;
03 |
     drop table LesGardiens ;
04 |
     drop table LesEMployes ;
05 |
     drop table LesCages ;
07 I
08 I
     create table LesCages (
09 |
             noCage number (3),
              fonction varchar2(20),
10 l
11 |
              noAllee number(3),
12 |
              constraint pk_LesCages_noCage PRIMARY KEY (noCage),
              constraint ck_LesCages_noCage CHECK(noCage > 0 AND noCage < 1000),</pre>
13 I
14 I
              constraint ck_LesCages_noAllee CHECK(noAllee > 0 AND noAllee < 1000)</pre>
15 l
     );
16 I
17 I
     create table LesEmployes (
18 I
              nomE varchar2(20)
19 |
              adresse varchar2(20),
              constraint pk_LesEmployes_nomE PRIMARY KEY (nomE)
20 |
21 I
     );
22 I
23 I
     create table LesGardiens (
24 |
              noCage number (3),
25 I
             nomE varchar2(20),
              constraint pk_LesGardiens_noCage_nomE PRIMARY KEY (noCage, nomE),
26 |
27 |
              constraint fk_LesGardiens_nomE FOREIGN KEY (nomE) REFERENCES LesEmployes (nomE
         ),
              constraint fk_LesGardiens_noCage FOREIGN KEY (noCage) REFERENCES LesCages (
28 I
         noCage),
29 I
              constraint ck_LesGardiens_noCage CHECK(noCage > 0 AND noCage < 1000)</pre>
30 |
     );
31 I
     create table LesResponsables (
32 |
33 I
             noAllee number(3),
34 |
              nomE varchar2(20),
              constraint pk_LesResp_noAllee_nomE PRIMARY KEY (noAllee, nomE),
35 I
36 I
              constraint fk_LesResp_nomE FOREIGN KEY (nomE) REFERENCES LesEmployes (nomE),
              constraint ck_LesResponsables_noAllee CHECK(noAllee > 0 AND noAllee < 1000)</pre>
37 I
38 I
     );
39 I
40 |
     create table LesAnimaux (
41 I
              nomA varchar2(20),
42 |
             sexe varchar2(13),
             type_an varchar2(15),
43 I
44 |
              fonction_cage varchar2(20),
              pays varchar2(20),
45 I
46 I
              anNais number (4),
47 I
              noCage number (3),
              constraint pk_LesAnimaux_nomA PRIMARY KEY (nomA),
48 I
49 |
              constraint fk_LesAnimaux_noCage FOREIGN KEY (noCage) REFERENCES LesCages (
         noCage),
50 I
              constraint ck_LesAnimaux_sexe CHECK (sexe IN ('femelle', 'male', '
          hermaphrodite')),
              constraint ck_LesAnimaux_noCage CHECK (noCage > 0 AND noCage < 1000),</pre>
51 l
              constraint ck_LesAnimaux_annais CHECK (annais >= 2000)
52 I
53 I
     );
54 I
55 I
     create table LesMaladies (
56 I
             nomA varchar2(20),
57 I
              nomM varchar2(20),
58 |
              constraint pk_LesMaladies_nomA_nomM PRIMARY KEY (nomA, nomM),
              constraint fk_LesMaladies_nomA FOREIGN KEY (nomA) REFERENCES LesAnimaux (nomA)
59 I
60 | );
```