SQL - Cours 4
Le langage SQL (Suite)

Ikbel GUIDARA

ikbel.guidara@univ-lyon1.fr
7/11/2017

Opérateur EXISTS

Opérateur (EXISTS)

- Permet de vérifier si le résultat d'une sous-requête est vide ou pas

```
SELECT Colonnes FROM Table WHERE EXISTS | NOT EXISTS (Sous-requête)
```

- La condition WHERE EXISTS (sous-requête)

  - est **vraie** si la sous-requête renvoie au moins un tuple
  - est **fausse** sinon

- La condition WHERE NOT EXISTS (sous-requête)

  - est **vraie** si la sous-requête ne renvoie aucun tuple
  - est **fausse** sinon

Opérateur (EXISTS)

le produit le plus léger

```
SELECT NP FROM P P1  WHERE
        NOT EXISTS (SELECT * FROM P P2 WHERE P1.Poids > P2.Poids)
```

le produit qui n'est pas le plus léger

```
SELECT NP FROM P P1  WHERE
        EXISTS (SELECT * FROM P P2 WHERE P1.Poids > P2.Poids)
```

les numéros d'usines qui ne reçoivent aucun produit rouge

```
SELECT NU FROM U WHERE
        NOT EXISTS (SELECT * FROM P, PUF WHERE P.NP=PUF.NP and
        couleur='rouge' and PUF.NU=U.NU)
```

La norme SQL 92

Jointure

- Jointure entre deux tables (Table 1 et Table 2)

```
SELECT Colonnes FROM Table 1 JOIN Table 2 ON Conditions

SELECT Colonnes FROM Table 1 INNER JOIN Table 2 ON Conditions
```

Liste des employés avec le nom du département où ils travaillent

```
SELECT ename, dname FROM emp JOIN dept ON emp.deptno = dept.deptno

SELECT ename, dname FROM emp INNER JOIN dept ON emp.deptno = dept.deptno
```

- Ces 2 requêtes sont équivalentes à :

```
SELECT ename, dname FROM EMP, DEPT WHERE EMP.deptno = DEPT.deptno
```

Produit cartésien

```
SELECT Colonnes FROM Table 1 CROSS JOIN Table 2
```

```
SELECT ename, dname FROM emp CROSS JOIN dept
```

- Ceci est équivalent à :

```
SELECT ename, dname FROM EMP, DEPT
```

Jointure naturelle

- Equi-jointure entre les attributs de même nom. Les attributs de même nom seront présentés une seule fois.

```
SELECT Colonnes FROM Table 1 NATURAL JOIN Table 2
```

```
SELECT * FROM emp NATURAL JOIN dept
```

Jointure naturelle

- Ne pas préfixer les colonnes par les tables qui les contiennent dans le cas d'une jointure naturelle
- La requête suivante provoque une erreur :

```
SELECT ename, dname, DEPT.deptno FROM emp NATURAL JOIN dept
```

- Il faut écrire:

```
SELECT ename, dname, deptno FROM emp NATURAL JOIN dept
```

Jointure naturelle

- Jointure naturelle sur une partie seulement des colonnes communes
- Il faut utiliser la clause JOIN USING (s'il y a plusieurs colonnes, le séparateur de colonnes est la virgule)

```
SELECT Colonnes FROM Table 1 JOIN Table 2 USING (Colonne)
```

```
SELECT ename, dname FROM emp JOIN dept USING (deptno)
```

- Ceci est équivalent à :

```
SELECT ename, dname FROM emp NATURAL JOIN dept
```

Jointure externe

```
SELECT Colonnes FROM Table 1 RIGHT | LEFT OUTER JOIN Table 2 ON Condition
```

Liste des départements (numero et nom) et de leurs employés

```
SELECT DEPT.deptno, dname, ename FROM DEPT JOIN EMP ON DEPT.deptno = EMP.deptno
```

Un département qui n'a pas d'employé n'apparaîtra jamais dans le résultat.

On pourrait pourtant désirer une liste des divers départements, avec leurs employés s'ils en ont, sans omettre les départements sans employés.

Jointure externe

```
SELECT DEPT.deptno, dname, ename FROM emp RIGHT OUTER JOIN dept
ON emp.deptno = dept.deptno
```

RIGHT indique que la table dans laquelle on veut afficher toutes les lignes (la table dept) est à droite de OUTER JOIN.

```
SELECT DEPT.deptno, dname, ename FROM emp LEFT OUTER JOIN dept
ON emp.deptno = dept.deptno
```

LEFT indique que la table dans laquelle on veut afficher toutes les lignes (la table emp) est à gauche de OUTER JOIN.

Fonction de choix (CASE)

- Une fonction de choix existe dans la norme SQL2
- Elle correspond à la structure switch du langage C
- Elle remplace avantageusement la fonction decode d'Oracle

- Il existe **deux syntaxes** pour **CASE** :
  - Une qui donne une valeur suivant des **conditions** quelconques
  - Une qui donne une valeur suivant la **valeur d'une expression**

Fonction de choix (CASE)

Syntaxe 1

```
CASE
      WHEN condition1 THEN expression1
      [WHEN condition2 THEN expression2]
      …
      [ELSE expression_défaut]
END
```

Syntaxe 2

```
CASE expression
      WHEN valeur1 THEN expression1
      [WHEN valeur2 THEN expression2]
      …
      [ELSE expression_défaut]
END
```

```
SELECT ename,
  CASE
       WHEN (JOB= 'PRESIDENT') THEN 1
       WHEN (JOB= ‘MANAGER') THEN 2
       ELSE 3
   END  as Niveau
FROM emp;
```

```
SELECT ename,
  CASE JOB
       WHEN 'PRESIDENT' THEN 1
       WHEN ‘MANAGER’ THEN 2
       ELSE 3
   END  as Niveau
FROM emp;
```

Liste des employés avec leur catégorie (président = 1, Manager= 2, autre = 3), en appelant la colonne Niveau