SQL - 2

REQUÊTES SELECT SUR PLUSIEURS TABLEAUX

STUD

MATR	NUME	AN	GRUPA	DATAN	LOC	TUTOR	PUNCTAJ	CODS
1456	GEORGE	4	1141A	12-MAR-82	BUCURESTI		2890	11
1325	VASILE	2	1122A	05-OCT-84	PITESTI	1456	390	11
1645	MARIA	3	1131B	17-JUN-83	PLOIESTI		1400	11
3145	ION	1	2112B	24-JAN-85	PLOIESTI	3251	1670	21
2146	STANCA	4	2141A	15-MAY-82	BUCURESTI		620	21
3251	ALEX	5	2153B	07-NOV-81	BRASOV		1570	21
2215	ELENA	2	2122A	29-AUG-84	BUCURESTI	2146	890	21
4311	ADRIAN	3	2431A	31-JUL-83	BUCURESTI		450	24
3514	FLOREA	5	2452B	03-FEB-81	BRASOV		3230	24
1925	OANA	2	2421A	20-DEC-84	BUCURESTI	4311	760	24
2101	MARIUS	1	2412B	02-SEP-85	PITESTI	3514	310	24
4705	VOICU	2	2421B	19-APR-84	BRASOV	4311	1290	24

SPEC si BURSA

CODS	NUME	DOMENIU
11	MATEMATICA	STIINTE EXACTE
21	GEOGRAFIE	UMANIST
24	ISTORIE	UMANIST

TIP	PMIN	PMAX	SUMA
FARA BURSA	0	399	
BURSA SOCIALA	400	899	100
BURSA DE STUDIU	900	1799	150
BURSA DE MERIT	1800	2499	200
BURSA DE EXCEPTIE	2500	9999	300

L'objectif

- Dans de nombreux cas, l'operateur veut obtenir un résultat qui contient des données qui sont stockées en deux ou plusieurs tableaux de la base de données, comme dans les exemples suivants:
- La liste avec les noms des étudiants et le nom de leur spécialisation. Les tableaux contenant ces données sont STUD (pour le nom de l'étudiant) et SPEC (pour le nom de la spécialisation).
- La liste avec le nom de l'étudiant (du tableau STUD) et le type de sa bourse (du tableau BURSA).
- La liste avec le nom de l'étudiant, le nom de sa spécialisation et le montant de sa bourse – les données sont reparties en trois tableaux

Jointure (eng: JOIN)

- L'opération qui permet une telle recherche est appelée jointure (eng.: join) et est effectuée via une requête SELECT avec les caractéristiques suivantes:
- Sur la clause FROM on met la liste de tableaux ou sont stockées les données.
- Sur la clause WHERE on met une condition reliant les tableaux de cette liste (condition appelée la condition de jointure).

SYNTAXE

```
SELECT [DISTINCT] liste expressions
  FROM liste tableaux
  WHERE condition de jointure
     AND conditions supplémentaires
-- puis GROUP BY, HAVING, ORDER BY
```

Faites attention!

- Lorsque la condition de jointure est absente, chaque ligne d'une tableau de la liste FROM est concaténée à chaque ligne des autres tableaux – le résultat est le produit cartésien des tableaux de FROM.
- Si la condition de jointure contient seulement des égalités, l' opération est appelée équijointure (eng.: equi-join). Dans d'autres cas, nous avons un non-équijointure.
- Si sur FROM nous avons plusieurs fois le même tableau, l'opération s'appelle auto-jointure (eng.: self join).

Autres considérations

- Si une ligne d'un tableau ne correspond pas par la condition de jointure avec aucune ligne d'autres tableaux, elle ne participe pas au résultat de la requête. On peut toutefois demander que cela soit pris en compte pour le résultat, entraînant ce qu'on appelle une jointure externe (anglais: outer join).
- Jusqu'à la version 9 la syntaxe pour la jointure externe en Oracle était différente de la norme ANSI. Depuis cette version on a ajoute les types de jointure de la norme SQL: 1999 (SQL-3), y compris le cross-join, jointure naturelle ou les différent variantes de jointure externe.

Produit cartésien

- La requête :

```
SELECT *
FROM STUD, SPEC;
```

retournera un résultat avec 12 colonnes et 36 lignes formées par la concaténation de chaque ligne de SPEC avec chaque ligne de STUD.

- Aussi, la requête:

```
SELECT DATAN, DOMENIU
FROM STUD, SPEC
WHERE LOC='BUCURESTI';
```

retournera un résultat avec 15 lignes

EQUIJOINTURE

```
SELECT MATR, STUD.NUME, STUD.CODS,
  SPEC.NUME
FROM STUD, SPEC
WHERE STUD.CODS = SPEC.CODS AND
DOMENIU='STIINTE EXACTE'
```

- On a marqué la condition de jointure. Le reste est la condition supplémentaire

NON-EQUIJOINTURE

```
SELECT NUME, AN, TIP, SUMA
FROM STUD, BURSA
WHERE PUNCTAJ BETWEEN PMIN AND PMAX
        AND CODS=11
```

- On a marqué la condition de jointure.

Alias pour un tableau

```
SELECT S.NUME, S.CODS, "SP
  STUD".NUME, TIP
FROM STUD S, SPEC "SP STUD", BURSA
WHERE S.CODS = "SP STUD".CODS AND
  S.PUNCTAJ BETWEEN PMIN AND PMAX
```

- L'alias doit répondre à certaines conditions (Oracle):

Alias pour un tableau

- Il ne peut pas être plus long que 30 caractères.
- Il doit commencer par une lettre, contenir que des lettres, des chiffres, _, # et $ ou il est placé entre des guillemets (aussi max. 30 caractères entre les guillemets).
- Entre les guillemets, les majuscules et les minuscules sont considérées comme des lettres différentes.
- Il ne peut être utilisé que dans la requête ou il a été introduit (il n'est pas stocke dans la base de données).

Alias pour un tableau

- Si un tableau a été associe avec un alias dans la clause FROM, l'alias peut et doit être utilisé pour préfixer les colonnes de ce tableau dans les autres clauses de la requête.
- Le nom du tableau ne peut plus être utilise pour préfixer dans ce cas.

Alias pour un tableau

- Parce que entre les guillemets les majuscules et les minuscules sont considérées comme des lettres différentes, la requete suivante renvoie une erreur:

```
SELECT S.NUME, STUD.CODS,"Sp Stud".NUME,
   TIP
FROM STUD S, SPEC "SP STUD", BURSA
WHERE S.CODS = "SP STUD".CODS;
```


Alias pour un tableau

- Aussi, si on veux préfixer avec le nom du tableau SPEC la requête renvoie une erreur (on doit utiliser l'alias):

```
SELECT S.NUME, STUD.CODS,SPEC.NUME, TIP
FROM STUD S, SPEC "SP STUD", BURSA
WHERE S.CODS = "SP STUD".CODS;
```


JOINTURE

- Si la condition de jointure n'est pas complète, le résultat contiendra un produit cartésien entre les jointures existantes et les autres tableaux.
- Exemple: dans la requête suivante la condition de liaison entre BURSA et les autres tableaux.
- Le résultat sera le produit cartésien entre BURSA et la jointure de STUD et SPEC (donc 5 lignes de BURSA x 12 lignes de la jointure STUD SPEC = 60 lignes).

```
SELECT ST.NUME, ST.CODS, SP.NUME, TIP
FROM STUD ST, SPEC SP, BURSA
WHERE ST.CODS = SP.CODS;
```


JOINTURE

1. Dans le cas général d'une jointure sur N tableaux, la condition de jointure contient (N – 1) sous-conditions reliés par AND (et logique) qui relient l'ensemble des tableaux.
2. En d'autres termes, si on construit un graphe où les nœuds sont les tableaux et les arcs sont les sous-conditions, le graphe doit être connecté.

Auto-jointure

- Exemple: la liste des étudiants et tuteurs.
- Il est obligatoire d'utiliser des alias:

```
SELECT S.NUME "NUME STUD",
       S.AN "AN STUD",
       T.NUME "NUME TUTOR",
       T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.TUTOR=T.MATR
```


Autre exemple

- Etudiants avec le même tuteur:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR;
```

- Erreur: on obtient des lignes comme (a, a)

Autre exemple

- Une autre variante:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR
      AND S1.MATR <> S2.MATR
```

- Erreur: on obtient des lignes comme (a, b) et (b, a) ensemble.

Autre exemple

- Pour obtenir le bon résultat:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR
      AND S1.MATR > S2.MATR
```


Jointure externe (Oracle)

- Pour ajouter les étudiants sans tuteur, on utilise la jointure externe:

```
SELECT S.NUME "NUME STUD", S.AN "AN
  STUD", T.NUME "NUME TUTOR",
  T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.CODS = 11 AND
      S.TUTOR=T.MATR(+)
```


Jointure externe (Oracle)

- Résultat:

```
NUME STUD  AN STUD NUME TUTOR AN TUTOR
---------- ------- ---------- --------
VASILE           2 GEORGE             4
GEORGE           4
MARIA            3
```

Jointure externe (Oracle)

- Si on change la place de (+) on obtient aussi les étudiants qui ne sont pas des tuteurs:

```
SELECT S.NUME "NUME STUD", S.AN "AN
  STUD", T.NUME "NUME TUTOR",
  T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.CODS = 11 AND
      S.TUTOR(+)=T.MATR
```


Jointure externe sans =

- La jointure externe peut exister aussi dans le cas ou la condition n'est pas une égalité. On peut utiliser par exemple <, <=, >, >= ou <>. La requête suivante retourne un résultat contenant 65 lignes:

```
SELECT S1.NUME "STUDENT 1", S1.PUNCTAJ
  "PUNCTAJ 1",
  S2.NUME "STUDENT 2",
  S2.PUNCTAJ "PUNCTAJ 2"
FROM STUD S1, STUD S2
WHERE S1.PUNCTAJ > S2.PUNCTAJ;
```


Jointure externe sans =

- Si on utilise la jointure externe, on obtient 66 lignes – la ligne supplémentaire est avec l'étudiant avec le plus grand nombre de points:

```
SELECT S1.NUME "STUDENT 1", S1.PUNCTAJ
  "PUNCTAJ 1",
  S2.NUME "STUDENT 2",
  S2.PUNCTAJ "PUNCTAJ 2"
FROM STUD S1, STUD S2
WHERE S1.PUNCTAJ (+) > S2.PUNCTAJ;
```


Jointure externe sans =

- Si on change le (+), on obtient aussi 66 lignes – la ligne supplémentaire est avec l'étudiant avec le plus petit nombre de points:

```
SELECT S1.NUME "STUDENT 1", S1.PUNCTAJ
  "PUNCTAJ 1",
  S2.NUME "STUDENT 2",
  S2.PUNCTAJ "PUNCTAJ 2"
FROM STUD S1, STUD S2
WHERE S1.PUNCTAJ > S2.PUNCTAJ(+);
```


Autres considérations

- On peut utiliser la jointure externe aussi dans le cas ou la condition de jointure est composée sauf le cas ou on a OR (ou logique) ou IN suivi par une liste contenant plusieurs valeurs.
- La jointure externe peut être utilise aussi avec les operateurs SQL.

Jointure externe + BETWEEN

- Les requête suivantes sont valides:

```
SELECT NUME, AN, TIP, SUMA
FROM STUD, BURSA
WHERE PUNCTAJ(+)-1000 BETWEEN PMIN AND
  PMAX

SELECT NUME, AN, TIP, SUMA
FROM STUD, BURSA
WHERE PUNCTAJ-1000 BETWEEN PMIN(+) AND
  PMAX(+)
```

Jointure externe + LIKE

- La requête suivante est valide :

```
SELECT S.NUME, S.CODS, F.NUME, F.CODS,
  SUBSTR(F.CODS, 2, 1)||'%' SABLON
FROM STUD S, SPEC F
WHERE S.CODS(+) LIKE SUBSTR(F.CODS, 2,
  1)||'%';
```


REZULTAT

NUME	CODS	NUME	CODS	SABLON
GEORGE	11	MATEMATICA	11	1%
VASILE	11	MATEMATICA	11	1%
MARIA	11	MATEMATICA	11	1%
GEORGE	11	GEOGRAFIE	21	1%
VASILE	11	GEOGRAFIE	21	1%
MARIA	11	GEOGRAFIE	21	1%
		ISTORIE	24	4%

JOIN EXTERN CU LIKE – cont.

- Aussi valide:

```
SELECT S.NUME, S.CODS, F.NUME, F.CODS,
       SUBSTR(F.CODS, 2, 1)||'%' SABLON
FROM STUD S, SPEC F
WHERE S.CODS LIKE SUBSTR(F.CODS(+), 2,
  1)||'%'
```


Les jointure SQL - 3

On utilise les clauses:

- CROSS JOIN – produit cartésien
- JOIN … USING – jointure sur les colonnes communes
- NATURAL JOIN – jointure naturelle
- JOIN … ON – jointure générale
- OUTER JOIN … ON – jointure externe

Sur une clause on met un seul tableau (aussi pour FROM)

CROSS JOIN

- Retourne le produit cartésien
- Syntaxe:

```
SELECT [DISTINCT] lista_de_expresii
FROM table1
CROSS JOIN table2;
```

- Exemplu:

```
SELECT S.NUME, F.NUME
FROM STUD S
CROSS JOIN SPEC F; -- 36 lignes
```


CROSS JOIN

- Pour obtenir une jointure on peut ajouter a CROSS JOIN une clause WHERE avec la condition de jointure:

```
SELECT S.NUME, DATAN, F.NUME, DOMENIU
FROM STUD S
CROSS JOIN SPEC F
WHERE S.CODS=F.CODS;
```

JOIN … USING

- On obtient une équijointure d'après les colonnes avec le même nom qui sont spécifiées en USING.
- Syntaxe:

```
SELECT [DISTINCT] lista_de_expresii
  FROM table1
  JOIN table2 USING (nume_coloane)
```


Exemple

- STUD et SPEC ont 2 colonnes avec le même nom: NUME et CODS. On peut spécifier une équijointure d'après CODS comme ca:

```
SELECT S.NUME, DATAN, F.NUME, DOMENIU
FROM STUD S
JOIN SPEC F USING (CODS);
```

- C'est pas nécessaire de préfixer les noms des colonnes dans la liste USING.

EXEMPLU – cont.

- On peut ajouter sur WHERE des conditions supplémentaires (par exemple LOC='BUCURESTI'):

```
SELECT S.NUME, DATAN, F.NUME, DOMENIU
FROM STUD S
JOIN SPEC F USING (CODS)
WHERE LOC='BUCURESTI';
```


NATURAL JOIN

- C'est une équijointure an utilisant toutes les colonnes communes.
- Syntaxe:

```
SELECT [DISTINCT] lista_de_expresii
FROM table1
NATURAL JOIN table2
```


NATURAL JOIN

- Exemplu:

```
SELECT NUME, DATAN, DOMENIU
FROM STUD
NATURAL JOIN SPEC;
  -- echijoin dupa NUME si CODS
```

- Le résultat sera vide – sauf le cas ou un étudiant a le même nom comme sa spécialisation :)

JOIN .. ON

- C'est la jointure générale. La condition de jointure est spécifiée par ON. On peut ajouter WHERE avec des conditions supplémentaires ou on peut les mettre sur ON aussi.
- Syntaxe:

```
SELECT [DISTINCT] lista_de_expresii
  FROM table1
  JOIN table2 ON (expresie_logica)
```

Exemple

◆Exemple:

```
SELECT S.NUME, DATAN, F.NUME, DOMENIU
FROM STUD S
JOIN SPEC F ON (S.CODS=F.CODS AND
  LOC='PLOIESTI');
```

ou

```
SELECT S.NUME, DATAN, F.NUME, DOMENIU
FROM STUD S
JOIN SPEC F ON (S.CODS=F.CODS)
WHERE LOC='PLOIESTI';
```


Exemple

◆Le résultat:

```
NUME       DATAN     NUME       DOMENIU
---------- --------- ---------- ---------------
MARIA      17-JUN-83 MATEMATICA STIINTE EXACTE
ION        24-JAN-85 GEOGRAFIE  UMANIST
```


JOIN .. ON – cont.

◆La condition de JOIN .. ON est générale – on peut utiliser tous les opérateurs:

```
SELECT NUME, PUNCTAJ, TIP, SUMA
FROM STUD
JOIN BURSA ON (PUNCTAJ BETWEEN PMIN
  AND PMAX)
```


OUTER JOIN … ON

◆Pour la jointure externe la syntaxe est:

```
SELECT [DISTINCT] lista_de_expresii
FROM table1
LEFT   \
RIGHT   | OUTER JOIN table2
FULL   /
ON (table1.nume_coloana1 =
    table2.numecoloana2)
```


LEFT …

- Pour la jointure externe gauche (LEFT OUTER JOIN), les valeurs nulles viennent du tableau 2.
- Exemple:

```
SELECT S.NUME "NUME STUD", S.AN
  "AN STUD",T.NUME "NUME TUTOR",
  T.AN "AN TUTOR"
FROM STUD S
LEFT OUTER JOIN STUD T
ON (S.TUTOR=T.MATR);
```


LEFT …

Le résultat a 12 lignes. Le même résultat peut être obtenu avec l'ancienne syntaxe comme ça:

```
SELECT S.NUME "NUME STUD", S.AN
   "AN STUD",T.NUME "NUME TUTOR",
   T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.TUTOR=T.MATR(+);
```

RIGHT ...

- Pour la jointure externe gauche (LEFT OUTER JOIN), les valeurs nulles viennent du tableau 1.
- Exemple:

```
SELECT S.NUME "NUME STUD",
   S.AN "AN STUD",
   T.NUME "NUME TUTOR",
   T.AN "AN TUTOR"
FROM STUD S
RIGHT OUTER JOIN STUD T
ON (S.TUTOR=T.MATR);
```


RIGHT ...

Le résultat a 13 lignes. Le même résultat peut être obtenu avec l'ancienne syntaxe comme ça:

```
SELECT S.NUME "NUME STUD",
   S.AN "AN STUD",
   T.NUME "NUME TUTOR",
   T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.TUTOR(+)=T.MATR;
```


FULL ...

- C'est un sorte de réunion entre les résultats de LEFT et RIGHT OUTER JOIN.
- Exemple:

```
SELECT S.NUME "NUME STUD",
   S.AN "AN STUD",
   T.NUME "NUME TUTOR",
   T.AN "AN TUTOR"
FROM STUD S
FULL OUTER JOIN STUD T
ON (S.TUTOR=T.MATR);
```


NOTE

- On ne peut pas spécifier la jointure externe complète avec l'ancienne syntaxe. Une requête qui contient:

```
WHERE S.TUTOR(+)=T.MATR(+)
```

- signale une erreur:

ORA-01468: a predicate may reference only one outer-joined table.