Niște salarii

Fiola de SQL (11) – *Marin Fotache*

Deși cu puternice conotații sindicale, și problema de luna aceasta este una preluată din folclorul SQL. Joe Celko o prezintă în cartea sa de "ghicitori" SQL la puzzle-ul cu nr. 15, unde îi și dezvăluie autorul, Jack Wells, cel care a plasat-o pe Compuserve în iunie 1996.

Se dă tabela SAL care conține istoricul salariilor pentru angajații firmei:

```
CREATE TABLE SAL (
   Marca INTEGER NOT NULL,
   DataModif DATE NOT NULL,
   Salariu DECIMAL (16) NOT NULL,
   PRIMARY KEY (Marca, DataModif)
   )
```

Orice modificare de salariu, în sensul creșterii sau, Doamne ferește!, micșorării (angajatul a fost retrogradat, nu mai este în grațiile șefului sau partidului de guvernământ, leul s-a apreciat exagerat de mult în raport cu dolarul etc.), presupune inserarea unei linii noi în care se indică marca fericitului/prea-nefericitului angajat, data modificării și noul cuantum al salariului.

Cu o asemenea tabelă putem reconstitui traseul salarial al unui angajat și chiar purcede la analize mai profunde, precum determinarea corelației dintre evoluția salariului, vârsta șefului și cea a secretarei sale. Din păcate, azi nu vom face lucruri atât de interesante, ci ne vom ocupa de o problemă mult mai prozaică: care sunt ultimele două salarii ale unui angajat, atât ca mărime, cât mai ales ca evoluție în timp ?

Care sunt cele mai mari două salarii ?

Flower-power la ureche !, cum ar zice informaticianul român din Canada sau SUA. Singura grijă ține de faptul că există angajați care nu prezintă, din momentul angajării lor până la cel al interogării, decât un singur cuantum. Așa încât este necesară joncțiunea externă. Iată soluțiile Oracle și DB2.

Oracle 8:

```
SELECT S1.Marca, MAX(S1.Salariu) SalMaxim,
MAX(NVL(S2.Salariu,0)) SalPrecedent
FROM SAL S1, SAL S2
WHERE S1.Marca=S2.Marca (+) AND
S1.DataModif > S2.DataModif (+)
GROUP BY S1.Marca
```

DB2 v7:

```
SELECT S1.Marca, MAX(S1.Salariu) SalMaxim,
MAX(COALESCE(S2.Salariu,0)) SalPrecedent
FROM SAL S1 LEFT OUTER JOIN SAL S2
ON S1.Marca=S2.Marca  AND S1.DataModif > S2.DataModif
GROUP BY S1.Marca
```

Se joncționează extern la stânga două instanțe ale tabelei SAL. Condiția de joncțiune este egalitatea atributului Marca și inegalitatea celor două date (calendaristice). Predicatul S1.DataModif > S2.DataModif garantează că din S1 se va extrage ultima dată a modificării, care este cea a intrării pe rol a salariului curent, în timp ce din S2 va fi extrasă o dată precedentă celei din S1. Funcțiile MAX combinate cu GROUP BY extrag ambele valori care ne interesează.

DB2 are implementată joncțiunea externă à la SQL/92, în timp ce Oracle-iștii se țin încă tare în privința asta și nu dau (+) pe OUTER JOIN. În plus, dacă înlocuim funcția COALESCE cu NVL, din soluția DB2 o obținem și pe cea Visual Fox Pro.

Care sunt cele mai mari trei salarii ?

Fiind o variantă a problemei prea puțin diferită de precedenta, prezentăm numai interogarea pentru Oracle 8:

```
SELECT S1.Marca, MAX(S1.Salariu) Ultimul,
MAX(NVL(S2.Salariu,0)) AS Penultimul,
MAX(NVL(S3.Salariu,0)) AS AntePenultimul
FROM SAL S1, SAL S2, SAL S3
WHERE S1.Marca=S2.Marca (+) AND S1.Salariu > S2.Salariu (+)
AND S2.Marca=S3.Marca (+) AND S2.Salariu > S3.Salariu (+)
GROUP BY S1.Marca
```

Care sunt cele mai recente două salarii și datele la care au intrat în vigoare ?

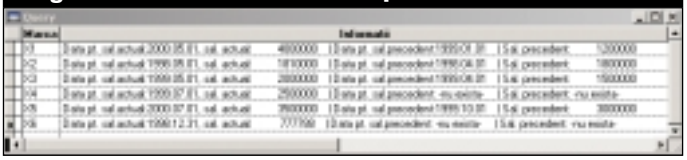
De fapt, chair asta e problema formulată de dom' Celko. Lucrurile se complică sensibil, deoarece, pentru fiecare dintre cele două salarii, acum ne interesează două informații, și data, și cuantumul. Pe lângă trimiterile la interogări formulate în lucrarea guru-ului, vom încerca și niște soluții mioritice, nu de alta, dar să-i arătăm lui Joe că SQL-iștii români sunt, și ei, în felul lor, *ortomani* (by the way, cum s-o fi zicând în English ?)…

Să începem cu o variantă căznită, valabilă pentru Visual FoxPro:

```
SELECT s1.Marca, ;
MAX('Data pt. sal.actual:'+;
   DTOC(s1.DataModif)+', sal. actual:'+;
   STR(s1.salariu,16)+;
   '| Data pt. sal.precedent:'+;
NVL(DTOC(s2.DataModif),' -nu exista- ')+;
   '| Sal. precedent:'+;
NVL(STR(s2.salariu,16), ' -nu exista- '))  ;
AS Informatii;
FROM salarii s1 LEFT OUTER JOIN salarii s2 ;
   ON s1.Marca=s2.Marca AND ;
s1.DataModif > NVL(s2.DataModif,{^1900/01/01})  ;
GROUP BY s1.Marca
```

La drept vorbind, nu avem prea multe motive de mândrie pentru așa o soluție. Ținând seama de fragilitatea VFP-ului în materie de SQL, bucuria este oarecum îndreptățită; la urma urmei, totuși, merge (și-așa)… Ideea interogării este de a constitui câte un grup pentru fiecare salariat și de a extrage, pentru fiecare grup/salariat, maximul dintr-un șir de caractere în care prima jumătate ar furniza data și cuantumul salariului actual, iar a doua dată și cuantumul salariului precedent. Datorită dispunerii celor patru atribute: data curentă, salariu curent, data precentă,

Figura 1. Prima variantă de răspuns în VFP



salariu precedent, funcția MAX extrage tocmai valorile care ne interesează, după cum se vede în figura 1.

Transcrierea în Oracle și DB2 poate fi făcută astfel:

Oracle:

```
SELECT S1.Marca, MAX(TO_CHAR(S1.DataModif,'YYYY/MM/DD')||' -
    '||
    TO_CHAR(S1.Salariu,'9999999999')) AS Data_Salariu_Ultim,
    MAX(NVL(TO_CHAR(S2.DataModif,'YYYY/MM/DD'),'    /  /
    ')||
    ' - '||NVL(TO_CHAR(S2.Salariu,'9999999999'),' '))
AS Data_Salariu_Penultim
FROM SAL S1, SAL S2
WHERE S1.Marca=S2.Marca (+) AND
S1.DataModif > S2.DataModif (+)
GROUP BY S1.Marca
```

DB2:

```
SELECT S1.Marca, MAX(CHAR(S1.DataModif, ISO)||' - '||
CHAR(S1.Salariu))
AS Data_Salariu_Ultim,
MAX(COALESCE(CHAR(S2.DataModif,ISO),'      /  /  ')||
' - '||COALESCE(CHAR(S2.Salariu),' '))
AS Data_Salariu_Penultim
FROM SAL S1 LEFT OUTER JOIN SAL S2
ON S1.Marca=S2.Marca AND
S1.DataModif > S2.DataModif
GROUP BY S1.Marca
```

Varianta propusă de Richard Romley la soluția nr. 5 din cartea lui Celko este mai elegantă, deși nu supărător de elegantă. Transcrisă pe calapodul VFP arată cam așa:

```
SELECT s1.Marca, ;
    MAX(s1.DataModif) AS Data_Sal_Actual, ;
    VAL(SUBSTR(MAX(DTOC(s1.DataModif)+;
STR(s1.Salariu,16)), 11, 16)) AS Sal_Actual, ;
    NVL(MAX(s2.DataModif), {..}) AS Data_Sal_Precedent, ;
    VAL(SUBSTR(MAX(DTOC(NVL(s2.DataModif,{  .  .  }))+;
STR(NVL(s2.Salariu,0),16)), 11, 16)) AS Sal_Precedent ;
FROM salarii s1 LEFT OUTER JOIN salarii s2 ;
    ON s1.Marca=s2.Marca AND ;
s1.DataModif > NVL(s2.DataModif,{^1900/01/01})  ;
GROUP BY s1.Marca
```

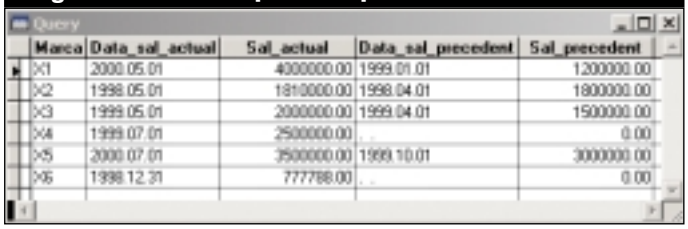
Cel mai mare câștig este modalitatea de prezentare a rezultatelor, după cum se vede în figura 2, informațiile fiind prezentate pe coloane distinctă. Fiecare grup lucrează mână în mână cu patru funcții MAX. Finețea soluției lui Romley este la extragerea salariilor. MAX-ul respectiv selectează tot cea mai mare dată calendaristică, însă, prin SUBSTR, se afișează numai salariul corespunzător.

Prin comparație cu următoarele variante, și aceasta are avantajul funcționării în VFP. Nu ne mai irosim însă timpul cu trecerea sa în formatele Oracle și DB2, pentru care vin vremuri (interogări) mai bune.

Să ne concentrăm pe o soluție dintre cele conforme cu SQL/92, bazate pe subconsultări în clauza FROM. Iat-o în sintaxa DB2:

```
SELECT ACTUAL.Marca, Data_Sal_Actual, Sal_Actual,
    Data_Sal_Precedent, Sal_Precedent
FROM
     (SELECT SAL.Marca, DATA1.Data AS Data_Sal_Actual,
          SAL.Salariu AS Sal_Actual
    FROM SAL INNER JOIN
          (SELECT Marca, MAX(DataModif) AS Data
          FROM SAL GROUP BY Marca) DATA1
    ON SAL.Marca=DATA1.Marca AND
SAL.DataModif=DATA1.Data)
ACTUAL
LEFT OUTER JOIN
    (SELECT SAL.Marca, DATA2.Data AS Data_Sal_Precedent,
    SAL.Salariu AS Sal_Precedent
    FROM SAL INNER JOIN
          (SELECT Marca, MAX(DataModif) AS Data
          FROM SAL
          WHERE (Marca, DataModif) NOT IN
                    (SELECT Marca, MAX(DataModif)
                         FROM SAL GROUP BY Marca)
          GROUP BY Marca) DATA2
                   ON DATA2.Marca=SAL.Marca AND
DATA2.Data = SAL.DataModif)
PRECEDENT
    ON  ACTUAL.Marca = PRECEDENT.Marca
```

Figura 2. Varianta optimă de prezentare a rezultatelor



Marca	Data_sal_actual	Sal_actual	Data_sal_precedent	Sal_precedent
X1	2000.05.01	4000000.00	1999.01.01	1200000.00
X2	1998.05.01	1810000.00	1998.04.01	1800000.00
X3	1999.05.01	2000000.00	1999.04.01	1500000.00
X4	1999.07.01	2500000.00	. .	0.00
X5	2000.07.01	3500000.00	1999.10.01	3000000.00
X6	1998.12.31	777788.00	. .	0.00

Încercăm o analiză top-down a interogării: aceasta se bazează pe joncțiunea externă la stânga a două *tabele ad-hoc* (să le spunem așa, dar să stabilim că este vorba de o improvizație, și nu de o contribuție la dezvoltarea științei), obținute prin subconsultări în clauza FROM, ACTUAL și PRECEDENT. La rândul ei, ACTUAL reprezintă joncțiunea dintre tabela-mumă SAL și o altă tabelă ad-hoc, DATA1. DATA1 conține câte o linie pentru fiecare angajat, având două coloane: marca și cea mai recentă DataModif. Prin joncțiunea dintre DATA1 și SAL pe baza condiției SAL.Marca=DATA1.Marca AND SAL.DataModif=DATA1.Data, ACTUAL va conține, pentru fiecare angajat: marca, data intrării în vigoare a salariului actual și cuantumul salariului actual. PRECEDENT se obține prin joncțiunea tabelei SAL cu tabela ad-hoc DATA2. Aceasta din urmă conține marca fiecărui angajat și penultima dată de modificare a salariului.

Interesantă mi se pare și soluția următoare, pe care v-o prezint urmând sintaxa Oracle 8:

```
SELECT S1.Marca, S1.DataModif AS Data_Sal_Actual,
    S1.Salariu AS Sal_Actual, S2.DataModif AS
    Data_Sal_Precedent,
    S2.Salariu AS Sal_Precedent
FROM SAL S1, SAL S2
WHERE  S1.Marca=S2.Marca (+) AND S1.DataModif > S2.DataModif (+)
AND S1.DataModif IN
(SELECT MAX(DataModif) FROM SAL WHERE Marca=S1.Marca)
AND NVL(S2.DataModif,TO_DATE('01-01-1900','DD-MM-YYYY')) IN
(SELECT MAX(DataModif)
                  FROM (SELECT * FROM SAL UNION
```

```
                SELECT DISTINCT Marca,
                TO_DATE('01-01-1900','DD-MM-YYYY'), 0
                                FROM SAL) SAL3
   WHERE SAL3.Marca=S1.Marca AND
DataModif < S1.DataModif)
```

Se joncționează extern la stânga două instanțe ale tabelei SAL, denumite SAL1 și SAL2, condiția de joncțiune fiind S1.Marca=S2.Marca AND S1.DataModif > S2.DataModif. Elementul de noutate al interogării este legat de faptul că, pentru fiecare angajat, cele cinci atribute sunt selectate fără a apela la GROUP BY, artificiul găsindu-se în clauza WHERE.

Acum trecem la prezentarea a două soluții DB2. Prima folosește expresiile-tabelă care sunt grozav de tentante pentru leneșii SQL:

```
WITH
   ACTUAL AS
        (SELECT SAL.Marca, DATA1.Data AS Data_Actuala,
   SAL.Salariu AS Salariu_Actual
         FROM SAL INNER JOIN (
   SELECT Marca, MAX(DataModif) AS Data
   FROM SAL GROUP BY Marca
      ) DATA1 ON SAL.Marca=DATA1.Marca AND
        SAL.DataModif=DATA1.Data),
   PRECEDENT AS
        (SELECT SAL.Marca, DATA2.Data AS Data_Precedenta,
   SAL.Salariu AS Salariu_Precedent
        FROM SAL INNER JOIN (
   SELECT Marca, MAX(DataModif) AS Data
   FROM SAL
   WHERE (Marca, DataModif) NOT IN
                  (SELECT Marca, MAX(DataModif)
                  FROM SAL GROUP BY Marca)
                        GROUP BY Marca
      ) DATA2 ON DATA2.Marca=SAL.Marca AND
        DATA2.Data = SAL.DataModif)
SELECT ACTUAL.Marca, Data_Actuala, Salariu_Actual,
   Data_Precedenta, Salariu_Precedent
FROM  ACTUAL LEFT OUTER JOIN PRECEDENT
ON  ACTUAL.Marca = PRECEDENT.Marca
```

Prin clauza WITH ce precede interogarea principală se crează două expresii-tabelă, în fond cu același regim ca tabelele ad-hoc dintr-o soluție anterioară.

A doua variantă DB2-specifică folosește inteogările scalare, o facilitate definită încă din SQL/92, dar neimplementată încă în multe SGBD-uri (precum Oracle):

```
SELECT Marca, MAX(DataModif) AS Data_Sal_Actual,
             (
          SELECT Salariu FROM SAL SAL2
          WHERE SAL2.Marca=SAL.Marca AND DataModif IN
             (SELECT MAX(DataModif)  FROM SAL SAL3
                WHERE Marca=SAL.Marca)
             ) Salariu_Actual,
             (
          SELECT MAX(DataModif)
          FROM SAL SAL2
          WHERE Marca=SAL.Marca AND DataModif <
   (SELECT MAX(DataModif) FROM SAL SAL3
   WHERE Marca=SAL.Marca)
          ) Data_Sal_Precedent,
             (
             SELECT Salariu
             FROM SAL SAL2
             WHERE SAL2.Marca=SAL.Marca AND DataModif IN
   (SELECT MAX(DataModif) FROM SAL SAL3
   WHERE Marca=SAL.Marca AND DataModif <
             (SELECT MAX(DataModif) FROM SAL SAL3
          WHERE Marca=SAL.Marca)
   )) Salariu_Precedent
FROM SAL
GROUP BY Marca
```

La modul simplist, interogările scalare pot fi definite ca fraze SELECT ce obțin o tabelă alcătuită dintr-o singură linie și o singură coloană. Deși par destul de stranii, utilizarea lor era extrem de necesară în multe situații care se rezolvă astăzi prin funcțiile OLAP ale SQL. Fără a intra în detalii, în exemplul nostru, ultimele trei coloane ale rezultatului (Salariu_Actual, Data_Sal_Precedent și Salariu_Precedent) sunt „calculate" prin interogări scalare. Prin clauza GROUP BY din fraza principală, rezultatul are câte o linie pentru fiecare angajat, iar pentru ca valoarea determinată de fiecare interogare corelată să fie cea corectă, se impune corelarea acesteia cu marca grupului curent.

No, asta a fost fiola pentru luna aceasta. Salarii să avem, că pe ultimele două știm cum să le aflăm…

Bibliografie

Celko, J. - Joe Celko's SQL Puzzles & Answers, Morgan Kaufmann, San Francisco, 1997

Marin Fotache este conferențiar la Catedra de Informatică Economică, UAIC Iași, Facultatea de Economie și Administrarea Afacerilor. Poate fi contactat pe email la: fotache@uaic.ro. ■ 76