Fiola de SQL (19)

Penalităţi pentru întârzierea plăţilor

Ancorat în sinergia faptelor, în fiola nr.17 – dedicată reducerilor acordate clienţilor pentru plata grabnică a facturilor – promiteam fierbinte că voi reveni în forţă cu o situaţie mult mai mioritică – cea a întârzierilor la plată şi calculul penalităţilor.

Cred că subiectul penalităţilor ar fi fost mai de actualitate luna următoare, deoarece decembrie este luna cadourilor, dar sper că îmi veţi scuza graba.

În plus, dacă în fiolele anterioare am privilegiat sintaxa PostgreSQL – aceasta şi pentru a semnaliza fanilor *free-database* faptul că m-am dat pe brazdă – azi le rezerv o dezamăgire: sintaxa frazelor SELECT va fi Oracle 8i. Nu de alta, dar cred că trecerea de la o sintaxă la alta este o operaţiune simplă, odată înţeleasă logica interogării.

Penalităţi pentru neplata la timp a facturilor

Probabil că vigilenţii cetitori au observat că în titlul paragrafului am folosit sintagma *la timp*, ceea ce înseamnă ok, în regulă. Cuvântul care precede *la timp*-ul este cel care induce anxietate şi răstoarnă situaţia cu 540 de grade – *neplata*.

Termenul de încasare (scadenţă) a unei facturi, precum şi procentul aplicat calculul penalizărilor pentru fiecare zi de întârziere (peste scadenţă) sunt reglementate prin contracte încheiate între firmă şi clienţi. Când nu există constract, termenul şi procentul pot fi specificaţi şi pe factură.

În schema bazei de date privind vânzările şi încasările, calculul penalizărilor necesită câteva atribute speciale. Spre exemplu, într-o tabelă CONTRACTE sau chiar în FACTURI pot fi introduse atributele DatăScadenţă şi ProcentPenalizări. Dacă se preferă prima variantă, este obligatoriu ca în tabela FACTURI să existe un câmp care să indice cărui contract îi este asociată factura de pe linia respectivă. A doua variantă, ce funcţionează chiar şi în situaţia inexistenţei unui contract prealabil, are în vedere includerea celor două atribute chiar în tabela FACTURI.

O a treia soluţie, mai complicată, dar şi mai flexibilă, presupune crearea unei tabele speciale al cărei nume poate fi CATEG_SCAD2 – de la CATEGORII_SCADENŢE, iar 2 (similar, mai ţineţi minte ?, DBase II) vrea să sugereze că exemplul este matur. Un cod desemnează o categorie de scadenţare, adică un număr de zile şi un procent pentru calculul penalizărilor pentru plăţi mai târzii decât numărul de zile scadente.

Fără a ne îndepărta prea mult de structura şi conţinutul tabelelor prezentate în fiola nr. 17, iată (în listing 1) comenzile de creare şi populare - de data aceasta specifice sintaxei Oracle.

Listing 1. Crearea în Oracle a tabelelor ce reflectă vânzările şi încasările

```
DROP TABLE incasfact2 ;
DROP TABLE incasari2 ;
DROP TABLE facturi2 ;
DROP TABLE categ_scad2 ;


CREATE TABLE categ_scad2 (
   CategScad NUMBER (4)
      CONSTRAINT pk_categ_scad2 PRIMARY KEY,
   ZileScadenta NUMBER(3),
```

```
   ProcPenZI NUMBER(5,2)
   ) ;

INSERT INTO categ_scad2 VALUES (1, 7, .15) ;
INSERT INTO categ_scad2 VALUES (2, 7, .5) ;
INSERT INTO categ_scad2 VALUES (3, 10, 1) ;
INSERT INTO categ_scad2 VALUES (4, 14, .25) ;


CREATE TABLE facturi2 (
   NrFact NUMBER(8)
      CONSTRAINT pk_facturi2 PRIMARY KEY,
   DataFact DATE DEFAULT SYSDATE,
   Client VARCHAR2(25),
   Valoare NUMBER(14),
   CategScad NUMBER(4)
      CONSTRAINT fk_facturi2_categ_scad2 REFERENCES categ_scad2(CategScad)
   ) ;

INSERT INTO facturi2 VALUES (1111, TO_DATE('01/08/2001', 'DD/MM/YYYY'), 'Client 1 SRL', 5399625, 1) ;
INSERT INTO facturi2 VALUES (1112, TO_DATE('01/08/2001', 'DD/MM/YYYY'), 'Client 5 SRL', 1337560, 1) ;
INSERT INTO facturi2 VALUES (1113, TO_DATE('01/08/2001', 'DD/MM/YYYY'), 'Client 2 SA',  1160250, 3) ;
INSERT INTO facturi2 VALUES (1114, TO_DATE('01/08/2001', 'DD/MM/YYYY'), 'Client 6 SA',  6786570, 2) ;
INSERT INTO facturi2 VALUES (1115, TO_DATE('02/08/2001', 'DD/MM/YYYY'), 'Client 1 SRL', 1651125, 3) ;
INSERT INTO facturi2 VALUES (1116, TO_DATE('02/08/2001', 'DD/MM/YYYY'), 'Client 7 SRL', 1383375, 1) ;
INSERT INTO facturi2 VALUES (1117, TO_DATE('03/08/2001', 'DD/MM/YYYY'), 'Client 1 SRL', 2320500, 4) ;
INSERT INTO facturi2 VALUES (1118, TO_DATE('04/08/2001', 'DD/MM/YYYY'), 'Client 1 SRL', 2052750, 3) ;
INSERT INTO facturi2 VALUES (1119, TO_DATE('07/08/2001', 'DD/MM/YYYY'), 'Client 3 SRL', 7242935, 1) ;
INSERT INTO facturi2 VALUES (1120, TO_DATE('07/08/2001', 'DD/MM/YYYY'), 'Client 1 SRL', 1066240, 2) ;
INSERT INTO facturi2 VALUES (1121, TO_DATE('07/08/2001', 'DD/MM/YYYY'), 'Client 4', 5438300, 4) ;


CREATE TABLE incasari2 (
   codinc NUMERIC(8)
      CONSTRAINT pk_incasari2 PRIMARY KEY,
--  in PostgreSQL:  datainc DATE DEFAULT CURRENT_DATE,
   datainc DATE DEFAULT SYSDATE,
   coddoc CHAR(4),
   nrdoc VARCHAR(16),
-- in PostgreSQL: datadoc DATE DEFAULT CURRENT_DATE - 7
   datadoc DATE DEFAULT SYSDATE - 7
   ) ;

INSERT INTO incasari2 VALUES (1234, TO_DATE('2001/08/15', 'YYYY/MM/DD'), 'OP', '111', TO_DATE('2000/08/10',
'YYYY/MM/DD'))  ;
INSERT INTO incasari2 VALUES (1235, TO_DATE('2001/08/15', 'YYYY/MM/DD'), 'CHIT', '222', TO_DATE('2000/08/15',
'YYYY/MM/DD'))  ;
INSERT INTO incasari2 VALUES (1236, TO_DATE('2001/08/16', 'YYYY/MM/DD'), 'OP', '333', TO_DATE('2000/08/09',
'YYYY/MM/DD') ) ;
INSERT INTO incasari2 VALUES (1237, TO_DATE('2001/08/17', 'YYYY/MM/DD'), 'CEC', '444', TO_DATE('2000/08/10',
'YYYY/MM/DD') ) ;
INSERT INTO incasari2 VALUES (1238, TO_DATE('2001/08/17', 'YYYY/MM/DD'), 'OP', '555', TO_DATE('2000/08/10',
'YYYY/MM/DD') ) ;
INSERT INTO incasari2 VALUES (1239, TO_DATE('2001/08/18', 'YYYY/MM/DD'), 'OP', '666', TO_DATE('2000/08/11',
'YYYY/MM/DD') ) ;
INSERT INTO incasari2 VALUES (1240, TO_DATE('2001/08/18', 'YYYY/MM/DD'), 'OP', '789', TO_DATE('2000/08/12',
'YYYY/MM/DD') ) ;


CREATE TABLE incasfact2 (
   codinc NUMERIC(8) CONSTRAINT fk_incasfact2_incasari2 REFERENCES incasari2(codinc) ,
   nrfact NUMERIC(8) CONSTRAINT fk_incasfact2_facturi2 REFERENCES facturi2(nrfact),
```

```
   transa NUMERIC(16) NOT NULL,
   CONSTRAINT pk_incasfact2 PRIMARY KEY (codinc, nrfact)
    ) ;

INSERT INTO incasfact2 VALUES (1234, 1111, 5399625)  ;
INSERT INTO incasfact2 VALUES (1234, 1118, 1026375)  ;
INSERT INTO incasfact2 VALUES (1235, 1112, 487705)  ;
INSERT INTO incasfact2 VALUES (1236, 1117, 975410)  ;
INSERT INTO incasfact2 VALUES (1236, 1118, 1026375)  ;
INSERT INTO incasfact2 VALUES (1236, 1120, 731557)  ;
INSERT INTO incasfact2 VALUES (1237, 1117, 975410)  ;
INSERT INTO incasfact2 VALUES (1238, 1113, 1160250)  ;
INSERT INTO incasfact2 VALUES (1239, 1117, 200000)  ;
INSERT INTO incasfact2 VALUES (1240, 1117, 169680)  ;

COMMIT ;
```

Cazul simplu (şi nerealist): facturile se încasează într-o singură tranşă

Dacă am porni de la premisa că toate facturile se încasează într-o singură tranşă, ar însemna că, în orice moment, există fie facturi încasate total, fie neîncasate deloc. Pentru facilitarea discuţiei, în toate interogările ce vor urma în această fiolă presupunem că data la care se calculează penalităţile este data curentă.

Interogarea de mai jos rezolvă problema reunind (UNION) penalizările facturilor deja încasate cu penalizările facturilor fără nici o încasare – sintaxa Oracle 8i:

```
SELECT f2.nrfact, datafact, valoare, datafact + zilescadenta AS data_scadenta,
   datainc, transa AS incasat,
   CASE WHEN datainc - datafact - zilescadenta <= 0
      THEN 0 ELSE datainc - datafact - zilescadenta END AS zile_penalizari,
   FLOOR (valoare *
   CASE WHEN datainc - datafact - zilescadenta <= 0
      THEN 0 ELSE datainc - datafact - zilescadenta END
       * procpenzi / 100 ) AS penalizare
FROM facturi2 f2, categ_scad2 cs2, incasari2 i2, incasfact2 if2
WHERE f2.categscad = cs2.categscad AND i2.codinc = if2.codinc
   AND f2.nrfact = if2.nrfact
UNION
   SELECT f2.nrfact, datafact, valoare, datafact + zilescadenta AS data_scadenta,
   SYSDATE, 0 AS incasat,
   CASE WHEN TRUNC(SYSDATE) - datafact - zilescadenta <= 0
      THEN 0 ELSE TRUNC(SYSDATE) - datafact - zilescadenta END AS zile_penalizari,
   FLOOR(valoare *
      CASE WHEN TRUNC(SYSDATE) - datafact - zilescadenta <= 0
         THEN 0 ELSE TRUNC(SYSDATE) - datafact - zilescadenta END
       * procpenzi / 100) AS penalizare
   FROM facturi2 f2, categ_scad2 cs2
   WHERE f2.categscad = cs2.categscad AND nrfact NOT IN
      (SELECT nrfact
       FROM incasfact)
```

Prin schimbarea constantei pentru data curentă SYSDATE şi alte câteva modificări minore, fraza poate fi trecută în orice sintaxă – PostgreSQL, DB2 etc.

Cazul trist (deci adevărat): facturile se încasează în oricâte tranşe

Cele patru tabele create şi populate în listingul 1 sunt mult mai apropiate de adevărul economic. Orice factură poate fi încasată în oricâte tranşe iar, pe de altă parte, cu un document de plată pot fi achitate mai multe facturi.

Cazul facturii 1117 este mai mult decât edificator, deoarece aceasta prezintă patru tranşe de încasare, deci, teoretic, pot exista patru componente ale penalităţilor plus o a cincea dacă factura nu este încă achitată integral.

Pentru orice factură:

- prima tranşă de încasare poate fi purtătoare de penalităţi dacă survine după data scadenţei şi se calculează la întreaga valoare a facturii;
- pentru a doua tranşă, eventualele penalităţi trebuie să vizeze numai zilele scurse de la prima tranşă şi diferenţa de plată (valoarea facturii minus prima tranşă de încasare);
- lucrurile se petrec aidoma pentru toate tranşele de încasare;
- dacă suma tranşelor este mai mică decât valoarea facturii, mai trebuie adăugată şi partea de penalităţi pentru numărul zile de la data ultimei tranşe până la data calculului penalităţilor aplicat la valoarea rămasă de încasat.

Încercând o abordare graduală a problemei să începem cu interogarea (Oracle):

```
SELECT incas2_1.nrfact, incas2_1.codInc, incas2_1.dataInc, incas2_1.transa,
   SUM(NVL(incas2_2.transa,0)) AS inc_preced,
   MAX(incas2_2.datainc) AS data_inc_preced,
   incas2_1.datainc -  NVL(MAX(incas2_2.datainc), incas2_1.datainc) AS zile_dif
FROM
   (SELECT nrfact, if2.codinc, datainc, transa
   FROM incasari2 i2, incasfact2 if2
   WHERE i2.codinc = if2.codinc) incas2_1,
       (SELECT nrfact, if2.codinc, datainc, transa
        FROM incasari2 i2, incasfact2 if2
      WHERE i2.codinc = if2.codinc) incas2_2
WHERE incas2_1.nrfact=incas2_2.nrfact (+) AND incas2_1.codinc > incas2_2.codinc (+)
GROUP BY incas2_1.nrfact, incas2_1.codinc, incas2_1.datainc, incas2_1.transa
```

care obţine rezultatul din figura 1, unde:

- fiecare linie se referă la o tranşă de încasare;
- inc_preced reprezintă suma tranşelor precente pentru factura respectivă;
- data_inc_preced este data tranşei precedente;
- zile_dif este numărul zilelor scurse între actuala tranşă şi precedenta.

NRFACT	CODINC	DATAINC	TRANSA	INC_PRECED	DATA_INC_PRECED	ZILE_DIF
1111	1234	15-AUG-01	5399625	0		0
1112	1235	15-AUG-01	487705	0		0
1113	1238	17-AUG-01	1160250	0		0
1117	1236	16-AUG-01	975410	0		0
1117	1237	17-AUG-01	975410	975410	16-AUG-01	1
1117	1239	18-AUG-01	200000	1950820	17-AUG-01	1
1117	1240	18-AUG-01	169680	2150820	18-AUG-01	0
1118	1234	15-AUG-01	1026375	0		0
1118	1236	16-AUG-01	1026375	1026375	15-AUG-01	1
1120	1236	16-AUG-01	731557	0		0

Figura 1. Calculul zilelor dintre două tranşe de încasare

Pentru a ne rezolva problema, trebuie să avem în vedere că pot exista şi facturi fără nici o încasare şi încasate parţial. De aceea, în tabela temporară incas2_1, creată într-o clauză FROM, mai adăugăm pentru fiecare factură şi câte o linie care reprezintă o tranşa ipotetică (codul încasării este maxim – 99999999, iar valoarea tranşei este zero) pentru data la care se calculează penalizarea – în cazul nostru chiar data curentă – SYSDATE.

Astfel modificată, interogarea de mai sus este inclusă într-o frază SELECT mamut:

```
SELECT f2.nrfact, f2.datafact, f2.valoare,
   f2.datafact + zilescadenta AS data_scadenta,
   transe.codinc, transe.datainc, NVL(transe.transa,0) AS transa,
   NVL(transe.inc_preced,0) AS inc_preced,
   transe.data_inc_preced, NVL(zile_dif,0) AS zile_dif,
   valoare - NVL(transe.inc_preced,0) AS dif_inc_preced,
   CASE WHEN datainc - NVL(data_inc_preced, f2.datafact + zilescadenta) <= 0
      THEN SYSDATE-SYSDATE
      ELSE datainc - NVL(data_inc_preced, f2.datafact + zilescadenta)
      END AS zile_pen,
   FLOOR ( (valoare - NVL(transe.inc_preced,0)) *
      (CASE WHEN datainc - NVL(data_inc_preced, f2.datafact + zilescadenta) <= 0
         THEN SYSDATE-SYSDATE
         ELSE datainc - NVL(data_inc_preced, f2.datafact + zilescadenta)
         END)
         * procpenzi  / 100 )AS penalizare
    FROM facturi2 f2, categ_scad2 cs2,
   (SELECT incas2_1.nrfact, incas2_1.codInc, incas2_1.dataInc, incas2_1.transa,
      SUM(NVL(incas2_2.transa,0)) AS inc_preced,
      MAX(incas2_2.datainc) AS data_inc_preced,
      incas2_1.datainc -  NVL(MAX(incas2_2.datainc), incas2_1.datainc)
         AS zile_dif
   FROM
      (SELECT nrfact, if2.codinc, datainc, transa
      FROM incasari2 i2, incasfact2 if2
      WHERE i2.codinc = if2.codinc
      UNION
```

```
        SELECT nrfact, 99999999 AS codinc, TRUNC(SYSDATE) AS datainc,
           0 AS transa
        FROM facturi2
    ) incas2_1,
        (SELECT nrfact, if2.codinc, datainc, transa
         FROM incasari2 i2, incasfact2 if2
        WHERE i2.codinc = if2.codinc) incas2_2
  WHERE incas2_1.nrfact=incas2_2.nrfact (+) AND
    incas2_1.codinc > incas2_2.codinc (+)
  GROUP BY incas2_1.nrfact, incas2_1.codinc,
    incas2_1.datainc, incas2_1.transa
  ) transe
WHERE f2.categscad=cs2.categscad AND f2.nrfact = transe.nrfact (+)
```

Rezultatul este cel din figura 2. Deşi datele numerice nu sunt aliniate la dreapta, formatul de prezentare este totuşi rezonabil.

NRFACT	DATAFACT	VALOARE	DATA_SCADENTA	CODINC	DATAINC	TRANSA	INC_PRECED	DATA_INC_PRECED	ZILE_DIF	DIF_INC_PRECED	ZILE_PEN	PENALIZARE
1111	01-AUG-01	5399625	08-AUG-01	1234	15-AUG-01	5399625	0		0	5399625	7	56696
1111	01-AUG-01	5399625	08-AUG-01	99999999	14-OCT-01	0	5399625	15-AUG-01	60	0	60	0
1112	01-AUG-01	1337560	08-AUG-01	1235	15-AUG-01	487705	0		0	1337560	7	14044
1112	01-AUG-01	1337560	08-AUG-01	99999999	14-OCT-01	0	487705	15-AUG-01	60	849855	60	76486
1113	01-AUG-01	1160250	11-AUG-01	1238	17-AUG-01	1160250	0		0	1160250	6	69615
1113	01-AUG-01	1160250	11-AUG-01	99999999	14-OCT-01	0	1160250	17-AUG-01	58	0	58	0
1114	01-AUG-01	6786570	08-AUG-01	99999999	14-OCT-01	0	0		0	6786570	67	2273500
1115	02-AUG-01	1651125	12-AUG-01	99999999	14-OCT-01	0	0		0	1651125	63	1040208
1116	02-AUG-01	1383375	09-AUG-01	99999999	14-OCT-01	0	0		0	1383375	66	136954
1117	03-AUG-01	2320500	17-AUG-01	1236	16-AUG-01	975410	0		0	2320500	0	0
1117	03-AUG-01	2320500	17-AUG-01	1237	17-AUG-01	975410	975410	16-AUG-01	1	1345090	1	3362
1117	03-AUG-01	2320500	17-AUG-01	1239	18-AUG-01	200000	1950820	17-AUG-01	1	369680	1	924
1117	03-AUG-01	2320500	17-AUG-01	1240	18-AUG-01	169680	2150820	18-AUG-01	0	169680	0	0
1117	03-AUG-01	2320500	17-AUG-01	99999999	14-OCT-01	0	2320500	18-AUG-01	57	0	57	0
1118	04-AUG-01	2052750	14-AUG-01	1234	15-AUG-01	1026375	0		0	2052750	1	20527
1118	04-AUG-01	2052750	14-AUG-01	1236	16-AUG-01	1026375	1026375	15-AUG-01	1	1026375	1	10263
1118	04-AUG-01	2052750	14-AUG-01	99999999	14-OCT-01	0	2052750	16-AUG-01	59	0	59	0
1119	07-AUG-01	7242935	14-AUG-01	99999999	14-OCT-01	0	0		0	7242935	61	662728
1120	07-AUG-01	1066240	14-AUG-01	1236	16-AUG-01	731557	0		0	1066240	2	10662
1120	07-AUG-01	1066240	14-AUG-01	99999999	14-OCT-01	0	731557	16-AUG-01	59	334683	59	98731
1121	07-AUG-01	5438300	21-AUG-01	99999999	14-OCT-01	0	0		0	5438300	54	734170

Figura 2. Calculul penalizărilor pentru fiecare tranşă de încasare

Facturile care nu au nici o încasare prezintă numai o linie în rezultat, iar celelalte câte o linie pentru fiecare tranşă, plus linia pentru ziua de calcul a penalizărilor. Structurile de tip CASE elimină eventualele valori negative ale numărului de zile de penalizare şi ale penalizărilor propriu-zise. Dacă pe ramura THEN am fi scris 0, cum era şi firesc, Oracle ar fi semnalizat o eroare legată de incompatibilitatea tipurilor de date, aşa încât apare expresia SYSDATE-SYSDATE care tot 0 înseamnă.

Rezultatul final, adică totalul penalizărilor pentru fiecare factură se obţine incluzând interogarea de mai sus într-un SELECT ce prezintă clauza GROUP BY:

```
SELECT nrfact, datafact, valoare, data_scadenta,
   SYSDATE AS data_penalizari, SUM(penalizare) AS penalizari
FROM
        (
... interogarea precedentă
          )
GROUP BY   nrfact, datafact, valoare, data_scadenta
```

Penalităţi către prea-fericitul şi relaxatul (fiscal) stat român

Calculul penalităţilor către stat au un cu totul alt regim şi sunt mai greu de extras din aplicaţii, dar fiind caracterul lor eterogen:

- TVA colectată – se poate determina din aplicaţia VÎNZĂRI;
- TVA deductibilă – dintr-o aplicaţie de gen FACTURI-INTRATE sau, după, caz din GESTOC, MIFIX s.a.;
- taxele vamale, accizele tot din module de genul celor de mai sus;
- impozitele şi contribuţiile la diverse fonduri (CAS, CASS, şomaj), delimitate pe destinaţii – din aplicaţia SALARIZARE;
- impozitul pe profit se calculează din module precum CONTABILITATE GENERALĂ sau BILANŢ.

Având în vedere cele de mai sus, cât şi datorită faptului că sunt un partizan (fără steaua roşie în cinci colţuri) al intervenţiei strict limitate a statului în economie, nu voi discuta astăzi acest subiect.

Pe curând !

Marin Fotache