＜会計計算を Stored Procedure でおこなう２ー薬剤＞

薬剤の点数は診療点数を求める場合とは異なり、単純ではありません。
薬剤は円単位で薬価が決められており、一定の規則に従って薬価を円表示から点表示に修正する必要があります。その場合、一つの薬剤ごとに点単位に変更するということであれば単純ですが、単位薬価という考え方があり、単位薬価を求めた後で点単位に変更することになります。
その例をあげますと、内服薬の単位薬価は次のように決められています。
①1 剤１日分をもとに点数にする
②単位薬価が１５円以下の場合は１点
③単位薬価が１５円を超える場合は１０円またはその端数を増すごとに１点を追加する
④複数の内服薬を処方する場合は服用時点及び服用回数がおなじものは１剤とする（例外あり）

最も重要なのは④の＜複数の内服薬を処方する場合は服用時点及び服用回数がおなじものは１剤とする＞というもので、複数の内服薬を処方した場合、投与日数・服用回数が同じ薬剤であれば１剤ということになるために、薬剤固有の薬価とは別に薬剤の組み合わせにより１剤としての単位薬価が定まるというものです。
このため、薬剤点数を求めるにはまず１剤ごとに単位薬価をもとめて、最終的に点数にするという作業が必要なために、はじめに集計値を複数行で抽出することが必須になります。

複数行を表示する Stored Procedure の構文は単一行を抽出する構文とはことなり、
FOR SELECT ...DO 文を使用します。
FOR SELECT ...DO 文では、
FOR SELECT で必要な行を抽出します。
DO では SELECT 文で抽出した行ごとに１度ずつ実行するブロックまたは文を記述します。

さて①から④の原則に基づいて薬剤点数を複数行で求める Stored Procedure は以下のようになります。

/* */で囲まれた文はコメントです。

```
------------------------------------------------------------------
/*薬剤点数を複数行で求める
*tani_ten は単位薬価で ten が求める点数
*/

CREATE PROCEDURE GET_YAKUZAI_TEN(KARTE_NO INTEGER)
RETURNS (naifukuyaku_ten DOUBLE PRECISION,nissuu INTEGER,tani_ten
INTEGER,ten INTEGER)
AS
DECLARE VARIABLE TANNI_YAKKA DOUBLE PRECISION;
DECLARE VARIABLE HINITI INTEGER;
```

```
BEGIN
FOR SELECT SUM(YAKKA*RYOU),MAX(DAYS)
FROM KARTE_DRUG
WHERE KARTE_NO=:karte_no
GROUP BY DAYS,TEKIYOU_CODE
INTO TANNI_YAKKA,HINITI
DO
BEGIN
nissuu=HINITI;
naifukuyaku_ten=TANNI_YAKKA;
IF (naifukuyaku_ten<=15) THEN
tani_ten=1;
ELSE
tani_ten=CAST(TANNI_YAKKA/10-1 AS INTEGER);
ten=nissuu*tani_ten;
SUSPEND;
END
END;
-------------------------------------------------------------------
---
```

（解説）
＊１行目の CREATE PROCEDURE GET_YAKUZAI_TEN(KARTE_NO INTEGER)の
(KARTE_NO INTEGER)が入力パラメーターです。
＊２行目の RETURNS (naifukuyaku_ten DOUBLE PRECISION,nissuu
INTEGER,tani_ten INTEGER,ten INTEGER)
の(naifukuyaku_ten DOUBLE PRECISION,nissuu INTEGER,tani_ten INTEGER,ten
INTEGER)が出力パラメーターです。
＊３・４行目の
DECLARE VARIABLE TANNI_YAKKA DOUBLE PRECISION;
DECLARE VARIABLE HINITI INTEGER;
は変数の宣言であり、変数の宣言は１行ごとに行います。
＊この StoredProcedure で最終的に必要な出力パラメーターは<ten>で、これが１剤ごとの点数です。
カルテ上にはこの点数が複数存在しますので、出力も複数になります。
最終的に１つのカルテ（１患者の受診１回分）の薬剤点数は複数出力された ten の合計になります。
＊下から５行目の tani_ten=CAST(TANNI_YAKKA/10-1 AS INTEGER);は CAST 関数が小数を四捨五入する機能にしたがっています。
＊FOR SELECT ...DO 文では、Stored Procedure の最初に宣言した変数に SELECT 内で FIELD 値を割り当て、最終的に DO 節以下で何らかの形で変数の値を出力パラメーターに渡すということになります。

実際に作成したStored Procedureの機能を確認します。
その前にまず、カルテ番号10131に入力されている薬剤データを抽出して確認します。
データベース上で以下のsql文を入力して実行します。

（入力）
```
SELECT * FROM KARTE_DRUG WHERE KARTE_NO=10131;
```

結果は以下のようになりました。
（結果）

	CODE	RYOU	DAYS	TEKIYOU_CODE	FUTAN_KUBUN	YAKKA	KARTE_NO
37	612140651	2	14	210H-3Epo	<null> H	100.6	10131
38	662640691	500	1	230H-Bpa	<null> H	35.5	10131
39	612170014	2	14	210H-3Epo	<null> H	29	10131
40	610408009	1	14	210H-3Apo	<null> H	3.16	10131
41	612180265	1	7	210H-3Apo	<null> H	174.4	10131

これによると、14日処方した薬剤でTEKIYOU_CODEとして同じものが2個あります。
したがって、結果は4グループであるので、結果の行は４行が出力されることになるはずです。
そこで実際に作成したStored Procedureを使用してカルテ番号10131内の薬剤点数を出力させます。

（入力）
```
SELECT * FROM GET_YAKUZAI_TEN(10131);
```

（結果）

NAIFUKUYAKU_TEN	NISSUU	TANI_TEN	TEN
17750	1	1774	1774
174.4	7	16	112
3.16	14	1	14
259.2	14	25	350

結果は確かに4行出力されました。単位薬価も正常に出力されています。

次に作成したStored Procedureからtenの合計を求めてみます。作成したStored Procedureはあたかもテーブルのように呼び出すことが出来ます。

（入力）
```
SELECT SUM(TEN) FROM GET_YAKUZAI_TEN(10131);
```
（結果）

```
       SUM
==========

      2250
```

確かに
1774+112+14+350=2250 になります。