はじめに

■コースの概要と目的

SQL を手続き型言語として機能拡張した Oracle 独自のプログラミング言語である PL/SQL について説明します。
PL/SQL の基本構文、ストアド・サブプログラムやトリガーの作成方法について実習を通して理解します。

■受講対象者

これから PL/SQL を使用してアプリケーション開発をされる方。

■前提条件

「SQL トレーニング」コースを受講された方。もしくは、同等の知識をお持ちの方。
※DML 文、トランザクションについてはすでに習得済みであること。

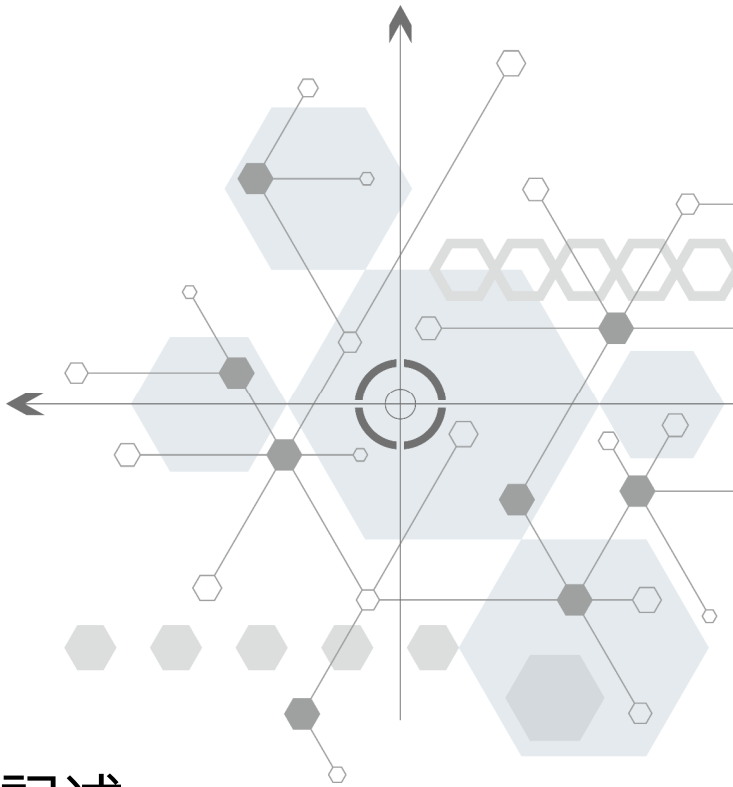
■テキスト内の記述について

▼構文

[ ]	省略可能
{ A | B }	A または B のどちらかを選択
n	数値の指定
_	デフォルト値

▼マーク

	指定バージョンからの新機能 （左記の場合、Oracle 19c からの新機能）
	Enterprise Edition で使用できる機能
	注意事項
	参考情報
	知っておくと便利なテクニック
	参照ページ
	データ・ディクショナリ・ビュー



第 2 章

PL/SQL の基本記述

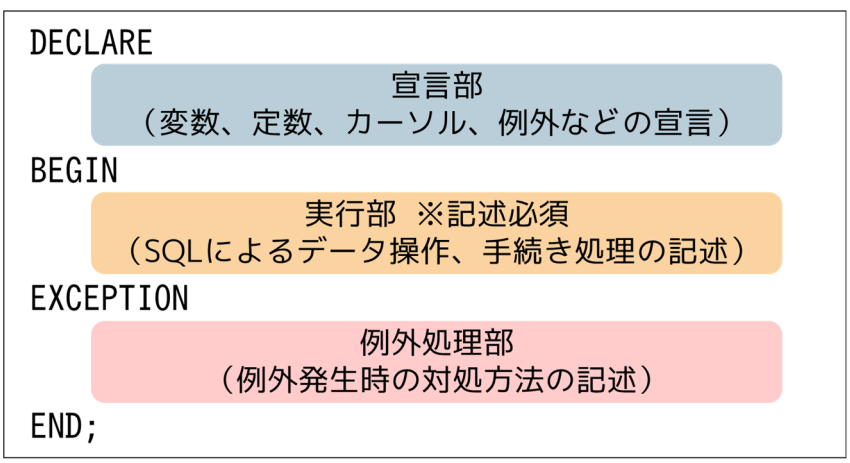
この章では、PL/SQL ブロックの基本的な記述方法について説明します。

01 宣言部の概要

宣言部では、実行部で使用するオブジェクトを定義します。必要でなければ省略することもできます。
宣言部は DECLARE キーワードで始まり、実行部の BEGIN キーワードにより暗黙的に終了します。
主に以下のオブジェクトを定義します。

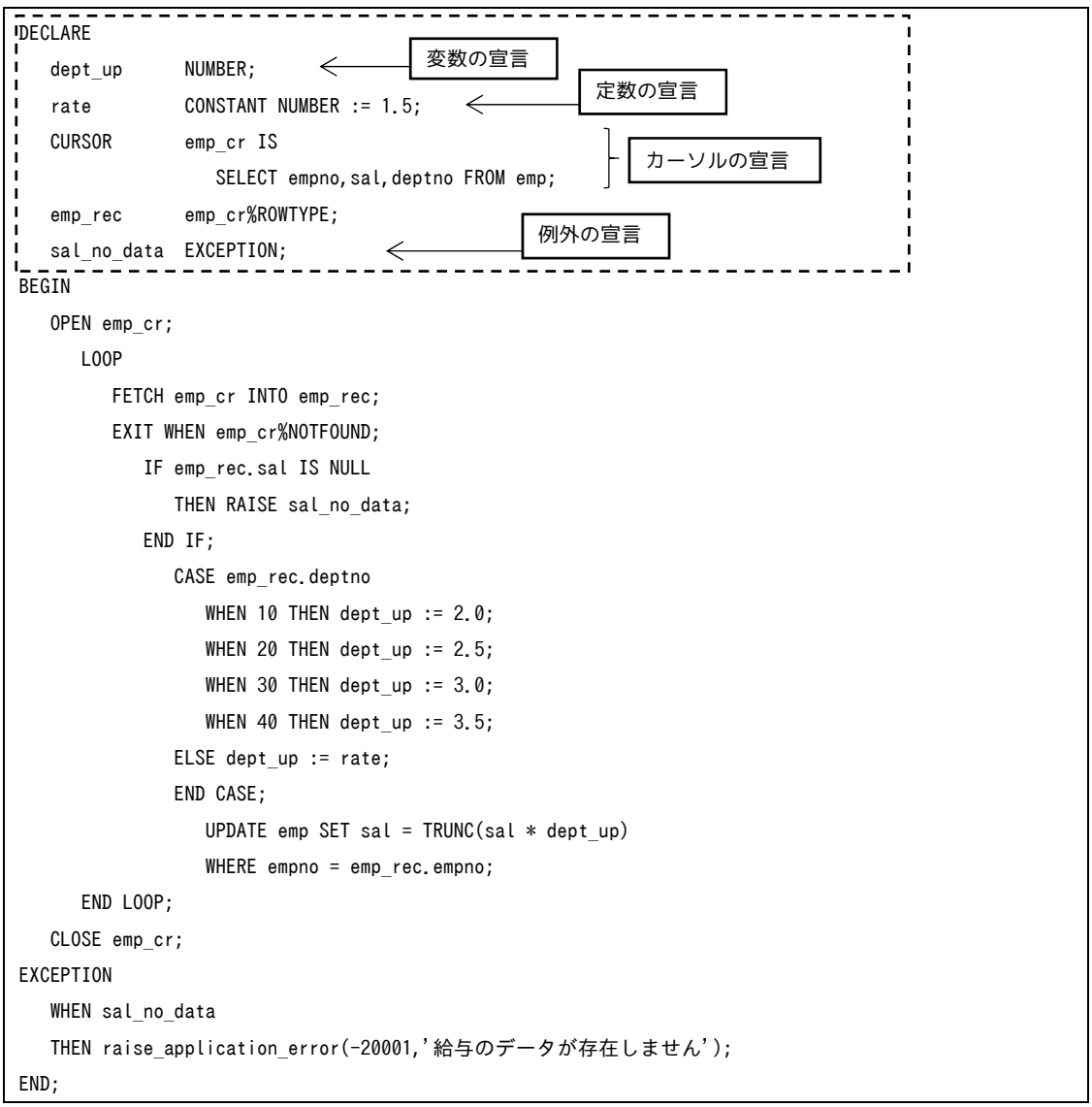
・変数
・定数
・カーソル
・例外
・コンポジット型

＜宣言部＞



Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.
株式会社 アシスト 無断転載・複製禁止

■宣言部（点線内）



株式会社 アシスト 無断転載・複製禁止
Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.

02 変数と定数

手続き型処理では、処理で使用する値を変数や定数に代入（格納）します。変数や定数などの名前（識別子）は宣言部で定義します。

（1）変数

変数とは、処理で使用する値を代入しておく場所です。処理の途中でデータベースのデータや計算結果を一時的に変数に代入できます。一度代入した変数に対して、異なる値を上書きして再利用できます。

構文

```
変数名 データ型 [ NOT NULL ] [ { := | DEFAULT } 値 ] ;
```

・代入演算子（:=）または DEFAULT キーワードを指定して初期値を設定できます。
　※デフォルトでは変数は NULL の状態です。
・NOT NULL キーワードを指定して変数に NULL が代入されないように定義することもできます。その際には変数に初期値を指定する必要があります。

（2）定数

定数とは、変数と同様、処理で使用する値を代入しておく場所です。ただし、変数と異なり定数は 1 つの決まった値を保持するため、定数に代入した値は上書きできません。そのため、プログラム内で固定値を扱う場合に使用します。

構文

```
定数名 CONSTANT データ型 { := | DEFAULT } 値 ;
```

・定数宣言時には必ず初期値を指定します。

変数のサイズ指定に算術式を指定できます。
例）ダブルバイト文字を最大 5 文字格納するための変数を定義する（二重引用符のために 2 バイト必要）。

```
var VARCHAR2 ( 10 + 2 ) ;
```

Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.　株式会社 アシスト 無断転載・複製禁止

例）変数、定数の宣言を行う。

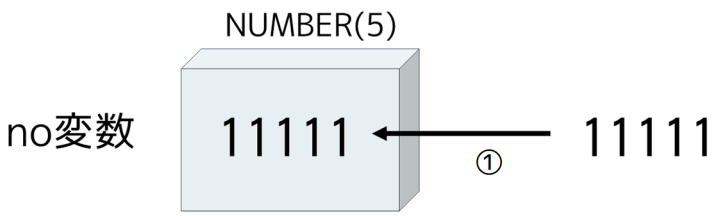
```
DECLARE
/* 変数 */
   no     NUMBER(5);
   name   VARCHAR2(10) := 'SMITH';
/* 定数 */
   prod   CONSTANT VARCHAR2(3) := 'ABC';
BEGIN
…省略…
```

数値型の値を代入できる no 変数を宣言

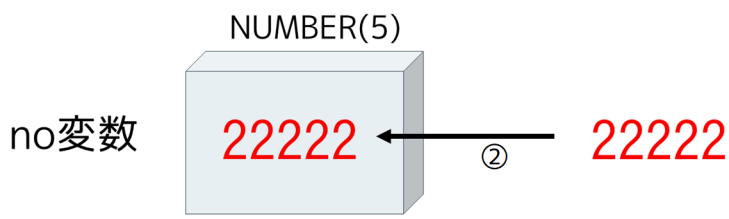
初期値を指定した name 変数を宣言

「ABC」という文字が代入された prod 変数を宣言
※値の上書きはできない

＜変数＞

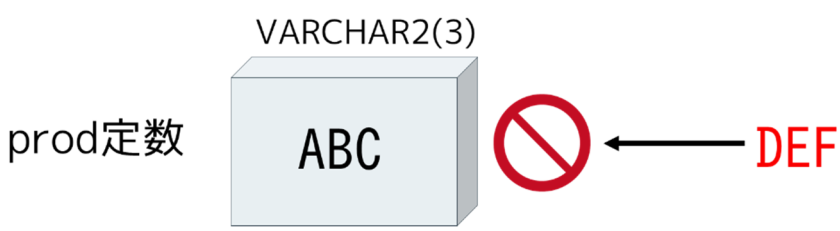


①no 変数に 11111 という数値データを代入。



②no 変数に 22222 という数値データを代入。
→11111 は上書きされる。

＜定数＞



prod 定数に DEF という文字データを代入。
→定数に対しては値を上書きできないため、エラーとなる。

株式会社 アシスト 無断転載・複製禁止
Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.

（3）%TYPE、%ROWTYPE 属性

%TYPE、%ROWTYPE 属性を使用すると、表の列や行、すでに宣言された変数のデータ型を参照することができます。参照元の定義が変更された場合でもコードを変更する必要がないため、主に表のデータを取り出して処理する場合に便利です。

1）%TYPE 属性

特定の表の列または既存の変数のデータ型を参照します。
※%TYPE 属性を使用した変数に初期値を指定することができます。

2）%ROWTYPE 属性

%ROWTYPE 属性は特定の表（またはビュー）の行構造を参照し、以下の特徴を持ちます。

・変数は表の行構造と同じ数のフィールドを持ちます。
・各フィールドの名前とデータ型には、参照表の列の名前とデータ型が対応づけられます。
・%ROWTYPE 属性を使用した変数は初期値を指定できません。
・各フィールドの参照は【変数名.フィールド名】で指定します。

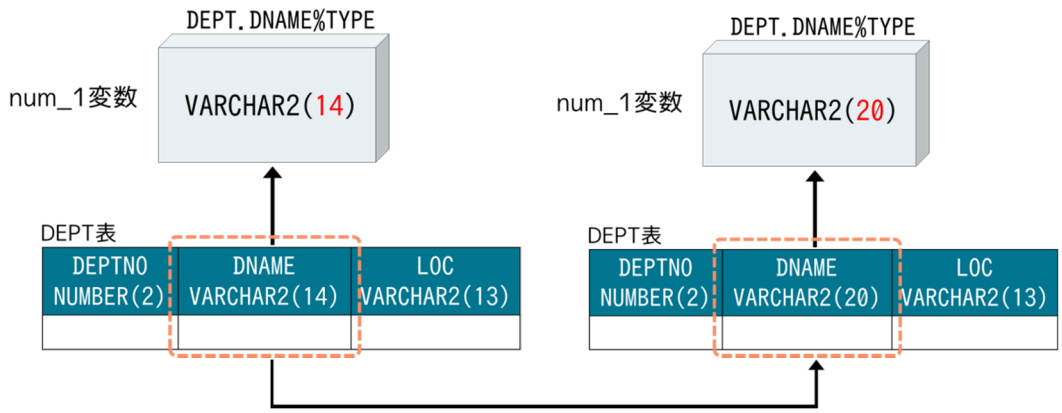
例）%TYPE、%ROWTYPE 属性を使用して変数を宣言する。

```
DECLARE
/* %TYPE 属性 */
   num_1   dept.dname%TYPE;  <――――――――― ①
   num_2   num_1%TYPE;    <――――――――― ②
   e_job   emp.job%TYPE := 'SALESMAN'; <―― ③
/* %ROWTYPE 属性 */
   d_rec   dept%ROWTYPE;   <――――――――― ④
BEGIN
…省略…
   UPDATE dept SET loc = d_rec.loc  <――― ⑤
   WHERE  deptno = 10;
…省略…
```

①num_1 変数は、DEPT 表の DNAME 列と同じデータ型で定義される。
②num_2 変数は、num_1 変数と同じデータ型で定義される。
③e_job 変数は、EMP 表の JOB 列と同じデータ型で定義され、初期値 SALESMAN が代入される。
④d_rec 変数は、DEPT 表の各列と同じデータ型、名前が各フィールドで定義される。
⑤d_rec 変数の LOC フィールドを参照するには、d_rec.loc と指定する。

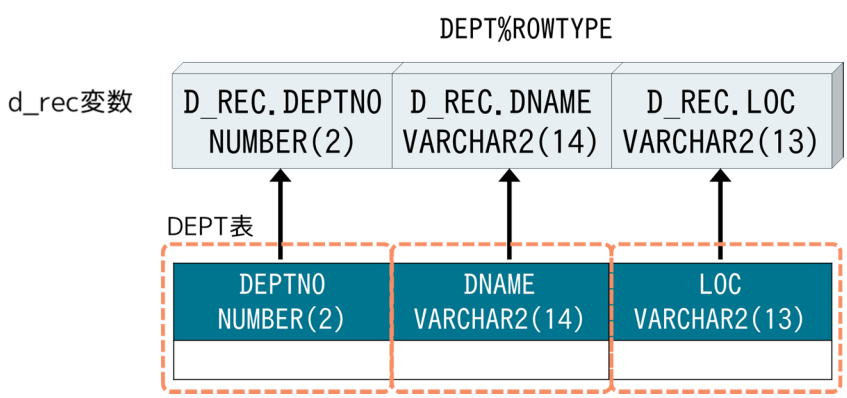
Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.　株式会社 アシスト 無断転載・複製禁止

＜%TYPE 属性＞



DEPT 表の DNAME 列を参照しているため、
変数の定義変更の必要がない。

＜%ROWTYPE 属性＞



各フィールドには DEPT 表の列と同じ名前、データ型が対応づけられる。

株式会社 アシスト 無断転載・複製禁止
Copyright (C) 2020 K.K.Ashisuto All Rights Reserved.