



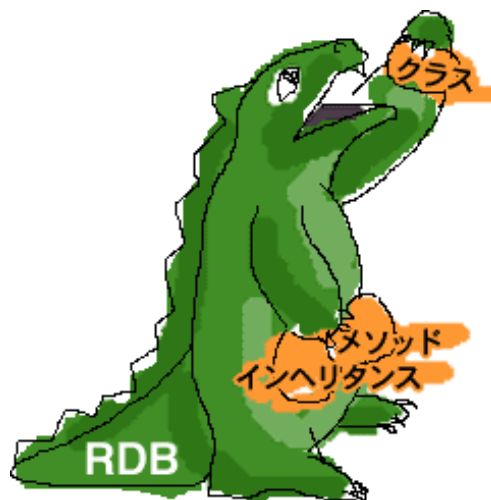
DB2いろはがるた



第33回

「こ」 - 構造化タイプ、SQL99のハイライト

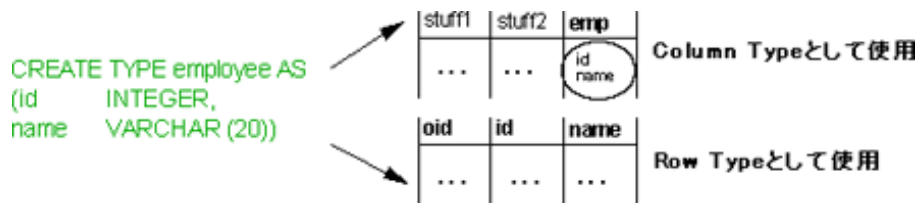
今回は構造改革ならぬ、SQLの構造化タイプのお話しをします。数年前に、データベースの発展の歴史の中である時期、オブジェクト指向プログラムの盛り上がり対応して、これからは既存のリレーショナル・データベース製品ではなくて、新しいオブジェクト・データベース製品の時代だと盛んにいわれたことがありました。その当時は、既存のリレーショナル・データベース製品を販売していた各ベンダーは、対オブジェクト・データベース製品への反論をしばしば行ったものです。あの熱気はどうなったのでしょうか？残念ながら、今は当時ほど盛り上がっているとは思えません。というのは既存の伝統的なリレーショナル・データベースが、新しいオブジェクト指向の考え方を標準で飲み込んでしまったからです。つまりSQL99でオブジェクト・リレーショナルという考え方が策定され、各リレーショナル・データベース製品が続々とその仕様の実装をはじめています。



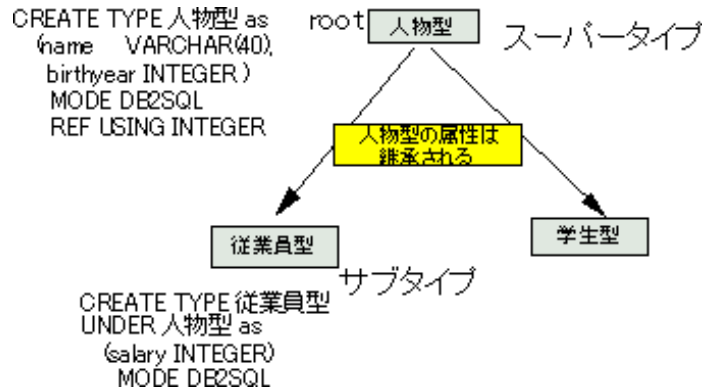
SQL99の標準で策定されたオブジェクト・データ型には次の3種類があります。

- 列型(COLUMN型)
- 行型(ROW型)
- 参照型(REF型)

構造型は表の1列として使用する列型 (COLUMN 型)と、表の1行の型として使用する行型 (ROW 型)があります。行型の定義型を表に適用したものを特にタイプ付き表といいます。他のユーザー定義構造型を参照することができるのが参照型です。



構造型はCreate data type文で定義し、次の図のように継承することができます。またその時に、typeに応じたメソッドを設定することもでき、それらのメソッドも継承できます。メソッドはリレーショナル・データベースの世界でいう特定のデータ・タイプだけに適用できる関数のようなものです。もし、タイプをクラスと考えれば、まさしくこの仕様はオブジェクト指向の考え方をリレーショナル・データベースにとりいれたものといえるでしょう。

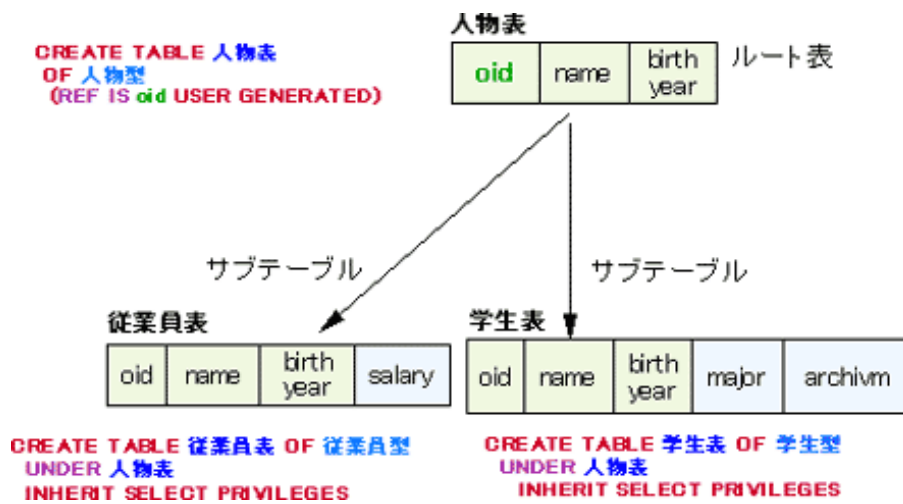


普通のデータ型と違って、行型の列にデータ値を挿入する場合には、特別なメソッド呼び出し演算子を使う必要があるので注意が必要です。下の例ではAddress列が構造化データ・タイプです。

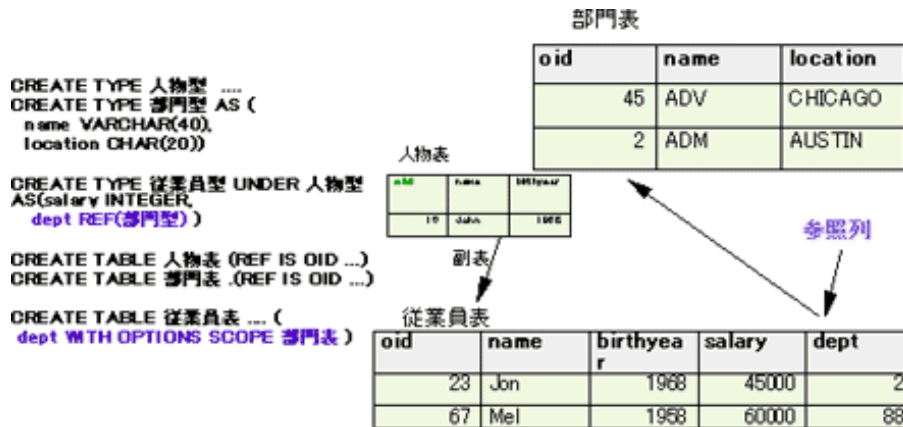
```
CREATE TABLE EMPLOYEE
(LASTNAME CHAR(20),
FIRSTNAME CHAR(20),
ADDRESS ADDRESS_T)
IN USERSPACE1;
```

```
INSERT INTO EMPLOYEE (LASTNAME, FIRSTNAME,
ADDRESS)
VALUES ('Hayato', 'Ichimonji',
US_ADDR_T()
..STREET ('123 Main Street')
..CITY ('Pukswane')
..STATE ('SD')
..ZIP ('70001'));
```

タイプ付き表とは、行型の構造型を基に作成した表のことで、ルートの構造型から作成した表をルート表、またサブタイプの構造型から作成した表をサブテーブル(副表)と呼びます。またサブテーブルの上位にあたる表をスーパーテーブル(上位表)と呼びます。



参照列とは他のユーザー定義構造型を参照すると宣言した、ユーザー定義構造型あるいはタイプ付き表の中の列のことです。各値はターゲット表の行で識別します。ここで、"ターゲット"表とは、参照先のタイプ付き表のことで、参照列は外部キーに類似していますが、同じではありません。参照列を通して、ターゲット表の任意の列値を得ることが可能です。



参照列を使うと、表の結合を次のようなSQL文で実行することができます。なんか矢印でSQL文を書くなで、不思議な気分ですね。

```
SELECT E.name from emp E
WHERE E.dept->location='AUSTIN'
```

構造化データのお話は少し複雑で頭がこんがらがったかも知れませんが、今後いろいろなデータ・タイプをリレーショナル・データベースに保管して管理しようとした場合、とても役に立つ機能となることでしょう。

[↑ 上に戻る](#)