ニューロ型データベースモデリング

小島　茂

Shigeru KOJIMA

「脳を構成する神経細胞（neuron）が繋がり合って情報を司る」という仕組みに着目し，リレーショナルデータベース管理システム(RDBMS)のテーブル設計にこのニューロ型を採用し，汎用性と拡張性を実現したデータベースモデリングを紹介する．

1. はじめに

この「ニューロ型データベースモデリング」は２個のテーブルだけで構築されている．

我々は，リレーショナルデータベース管理システム(RDBMS)を利用してあらゆるデータベースシステムを構築しているが，データベースシステムの作成には「データモデリング」が必須であり，データベース管理者は実体関連図（ER図）や統一モデリング言語（UML）等を駆使し，データモデリングに取り組み多くの開発時間を費やして構築している．

データモデリングの優劣は，そのデータベースシステムの優劣であると言っても過言ではない．

本論文では「脳の神経細胞（neuron）が繋がり合って情報を司る」という仕組みをRDBMSのテーブル設計に応用し「オブジェクト指向」の処理方法を実現したものである．

今までは新しい案件が発生する度に，新たなテーブルの作成，検索プログラムの変更等が必要であったが，このニューロ型データベースモデリングを利用すれば，データベースの設計は自由であり，登録したデータは簡単に移動やリンクが可能である．

検索プログラムは，データベース内のテーブルやその中の項目が増減しても変更の必要が無く，メンテナンスフリーである．

データベース内の検索は，まず「言葉」が登録されている列（表1）に対し検索キーワードを照合し，部分一致が無い場合はこれで検索が完了する．最初の検索で該当データの有無判定が非常に速いのが特徴である．

蓄積されていくデータは「ID（番号）」なので（表2），文字列データを直接保存していくよりも，少ないデータ量で構築が出来る為，非常にコンパクトな設計である．

「簡単」「速い」「コンパクト」な縦型データベースモデリングをこれからご紹介する．

2.「言葉」を覚え始める

2.1 Word Entity の作成

ニューロ型データベースモデリングで使用する「言葉」を保存していくテーブル「Word_Entity_tbl」を作成する（表1）．

- ［ID］： 0 から始まるプライマリキー
- ［言葉］：可変長文字列（重複なし）（Not Null）

2.2 Word Entity の入力規則

作成したテーブル「Word_Entity_tbl」の［ID］列に［0］を，［言葉］列に［なし］を登録，続いて［1］［未来］，［2］［現在］［3］［過去］を登録．知らない「言葉」が入力されたら直ぐに「Word_Entity_tbl」へ重複無しデータとして登録して行く．

- 氏名は「氏」と「名」を分けて登録

表1　Word Entity

ID	言葉
0	なし
1	未来
2	現在
3	過去
4	識別名
5	人
6	氏
7	氏ヨミ
8	名
9	名ヨミ
10	性別
11	西暦生年月日
12	E-mail
13	住所
14	小島
15	コジマ
16	茂
17	シゲル
18	男
19	1964-01-08
20	skojima@kitahama.or.jp
21	自宅
22	国名
23	〒
24	都道府県名
25	住所1
26	住所2
27	建物名
28	電話番号1
29	電話番号2
30	FAX番号
31	日本
32	近畿地方
33	関東地方
34	大阪府
35	東京都
36	栄作
37	エイサク
38	独立フォーム
39	中島
40	ナカジマ
41	大島
42	オオシマ
43	入力テンプレート
44	備考
※ 45	独立フォームなので

※備考等のデータは分割せず長い文字列のままで登録

3.「言葉」を繋げ始める

3.1 Identify Entity の作成

これからあらゆるオブジェクト（物体）を「識別化」し，ユニークな「ID」を付けて記憶していく（これ以降 **識別ID** と表す）テーブル「Identify_Entity_tbl」を作成する（表2）．

- ［主キー］： 1 から始まるプライマリキー
- ソート順：[識別ID][T_ID][ソート]
  ［識別ID][T_ID][ソート]は（重複あり）
- 全ての列は数値（型，Index等はRDBMS仕様に準拠）

表 2　Identify Entity

主キー	識別 ID	T_ID	ソート	ユニット	F_ID	D_ID	識別 ID リンク先
1	1	0	0	0	4	5	
2	1	2	1	1	6	14	
3	1	2	2	1	7	15	
4	1	2	3	2	8	16	
5	1	2	4	2	9	17	
6	1	2	5	0	10	18	
7	1	2	6	0	11	19	
8	1	2	7	0	12	20	
9	1	2	8	0	13	21	10
10	10	0	0	0	4	21	
11	10	2	1	1	22	31	
12	10	2	2	1	23		
13	10	2	3	1	24	34	
14	10	2	4	1	25		
15	10	2	5	1	26		
16	10	2	6	1	27		
17	10	2	7	0	28		
18	10	2	8	0	29		
19	10	2	9	0	30		

3.2 「識別 ID」化フォームの準備

まずは「人」を識別化する「入力フォーム」の準備をする．図1の入力フォーム（イメージ）を開くとバックグランドで以下の処理を実行する．

- ［主キー］： 1 （自動取得）
- ［識別ID］：**主キーと同じ値** 1 （自動登録）
- ［T_ID］： 0 （自動登録）
- ［時点］： なし （自動表示）
- ［ソート］： 0 （自動登録）手動登録も可
- ［ユニット］： 0 （自動登録）入力項目グループ化用

3.2.1 「識別名」の登録

図1は[入力項目]列に[識別名]，[入力データ]列に[人]が入力された状態である．

[F_ID]列に[4]，[D_ID]列に[5]が自動表示されているのは，テーブル「Word_Entity_tbl」の「言葉」列に登録が完了したからである．

- ［F_ID］： 4 （自動表示）
- ［入力項目］： 識別名
- ［D_ID］： 5 （自動表示）
- ［入力データ］： 人
- 登録ボタンを押し[識別ID]を確定

入力フォーム　　識別ID 1

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ
1	1	0	なし	0	0	4	識別名	5	人
*									

図 1 「識別名」の登録

3.2.2 「入力項目」の登録 －縦列連結－

「言葉」を[識別ID]で縦列連結（グループ化）すると[入力項目]となる．

図2は[入力項目]に手入力を終えた状態である．赤丸は[ユニット]であり，現在２つのグループが作成されている．今後[氏]を変更する時，同じユニット番号を持つ[氏ヨミ]も変更対象とする為に使用する．

- ［主キー］： ２ ３ ４ ５ ６ ７ （自動取得）
- ［識別ID］：**主キーと同じ値** 1 （自動登録）[1]
- ［T_ID］： 2 （自動登録）
- ［時点］： 現在 （自動表示）
- ［ソート］： 1 ～ 6 （自動連番）
- ［ユニット］： 0 （自動登録）後， １ １ ２ ２ を手入力
  （任意の整数で連番である必要はない）
- ［F_ID］： 6 7 8 9 10 11 （自動表示）
- ［入力項目］: 氏 氏ヨミ 名 名ヨミ 性別 西暦生年月日

入力フォーム　　識別ID 1

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ
1	1	0	なし	0	0	4	識別名	5	人
2	1	2	現在	1	1	6	氏		
3	1	2	現在	2	1	7	氏ヨミ		
4	1	2	現在	3	2	8	名		
5	1	2	現在	4	2	9	名ヨミ		
6	1	2	現在	5	0	10	性別		
7	1	2	現在	6	0	11	西暦生年月日		
*									

図2 「入力項目」の登録

3.3 「識別 ID」化フォームの入力－横列連結－

図3は[入力データ]にデータを入力した状態である．

- ［D_ID］： 14 15 16 17 18 19 （自動表示）
- ［入力データ］: 小島 コジマ 茂 シゲル 男 1964-01-08

[1] グループ管理番号である[識別 ID]は新しく識別名を追加する度に登録レコード先頭の[主キー]を採番するのでシーケンシャルな数値にはならず，この場合 1 の次は 10 になる

入力フォーム　　識別ID　1

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ
1	1	0	なし	0	0	4	識別名	5	人
2	1	2	現在	1	1	6	氏	14	小島
3	1	2	現在	2	1	7	氏ヨミ	15	コジマ
4	1	2	現在	3	2	8	名	16	茂
5	1	2	現在	4	2	9	名ヨミ	17	シゲル
6	1	2	現在	5	0	10	性別	18	男
7	1	2	現在	6	0	11	西暦生年月日	19	1964-01-08

図3　「識別 ID」化フォームの入力

［入力項目］と［入力データ］が横列連結すると「情報」となる．これで個人を特定する最低限の情報が揃ったので，[5][人]として識別化し[識別 ID][1]として登録する．

3.4 「識別 ID」化の完了

これで[識別 ID]が[1]である「人」の登録が完了した．

注目すべき点は全レコードの[識別 ID]に先頭の[主キー]列の[1]が自動登録された事である（手動入力は不可とする）．これにより縦型データベースで複数のデータをグループ化することが出来る．［入力項目］の追加はテーブルデザインで新しく列を作成しなくても良いので，いつでも簡単に［入力項目］を追加出来るのは便利な機能だ．この処理方法によりテーブルの列設計は一切変更しないので，メンテナンスフリーが実現する．

4.「拡張」を始める

4.1 入力項目の追加

人の[識別 ID]が確定したが同姓同名，同一生年月日，同一性別が２人以上存在する場合これだけでは「個人」の特定が出来ない．取りあえず個人情報として[識別 ID][1]に対し[入力項目]として[E-mail]を追加する（図 4）．

- ［主キー］： 8 （自動取得）
- ［識別ID］： 1 （自動登録）[1]
- ［T_ID］： 2 （自動登録）
- ［[時点］： 現在 （自動表示）
- ［ソート］： 7 （自動連番）
- ［ユニット］： 0 （自動登録）
- ［F_ID］： 12 （言葉登録処理後自動表示）
- ［入力項目］： E-mail
- ［D_ID］： 20 （言葉登録処理後自動表示）
- [入力データ]：skojima@kitahama.or.jp
- 登録ボタンを押しデータ追加確定

所有しているE-mailが１個とは限らないので2つ以上登録する場合は，もう１つ[E-mail]というレコードを追加しても良いし[E-mail2]という[入力項目]を新たに作成しても良い．

[1] グループ番号である[識別 ID]は新しく識別名を追加する度にレコード先頭の主キー値を利用するのでグループ番号（[識別 ID]）は１ずつ増加するシーケンシャルな数値にはならない

入力フォーム　　識別ID　1

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ
1	1	0	なし	0	0	4	識別名	20	人
2	1	2	現在	1	1	6	氏	14	小島
3	1	2	現在	2	1	7	氏ヨミ	15	コジマ
4	1	2	現在	3	2	8	名	16	茂
5	1	2	現在	4	2	9	名ヨミ	17	シゲル
6	1	2	現在	5	0	10	性別	18	男
7	1	2	現在	6	0	11	西暦生年月日	19	1964-01-08
8	1	2	現在	7	0	12	E-mail	20	skojima@kitahama.or.jp
9	1	2	現在	8	0	13	住所	21	自宅
*									

① ② ③ ソートキー

図4　入力項目の追加

- ［主キー］： 9 （自動取得）
- ［識別ID］： 1 （自動登録）
- ［T_ID］： 2 （自動登録）
- ［[時点］： 現在 （自動表示）
- ［ソート］： 8 （自動連番）
- ［ユニット］： 0 （自動登録）
- ［F_ID］： 13 （言葉登録処理後自動表示）
- ［入力項目］： 住所
- ［D_ID］： 21 （言葉登録処理後自動表示）
- [入力データ]：自宅
- 登録ボタンを押しデータ追加確定

［入力項目］だけ事前に登録しておきたい場合を考慮し，［入力項目］だけ登録されていて，［入力データ］が未登録のレコード存在も可とする．勿論，［入力項目］が設定されていない［入力データ］のみのレコードは登録不可とする（図4）．

5.「学習」を始める

5.1「識別 ID リンク先」の登録

個人を特定する話の続きである．「同姓同名，同一生年月日，同一性別であり E-mail 携帯電話がない」という場合，やはり個人を特定するには「自宅」「勤務先」が必要になる（図 5）．

- ［識別IDリンク先］：空欄をダブルクリック
- ［識別 ID リンク先］：識別 ID 番号（主キー番号）
  10 （自動登録）
- ［識別ID］[10] の入力フォーム画面へ移動

自宅を２つ以上所有している場合「自宅の数だけレコードを追加」することになる．

ここで、「人」以外の[識別 ID]を作成するのは初めてになるが，[入力項目]が異なるデータだが，登録先は「人」と同じテーブル「Word_Entity_tbl」なので，「人」の登録とまったく同じ手順で追加することになる．

今回は，［識別ID］[10]を新規で作成するが，これ以降は，［識別名］[21][自宅]で全体を検索し，[D_ID][入力データ]を削除したレコードを複製して使える．

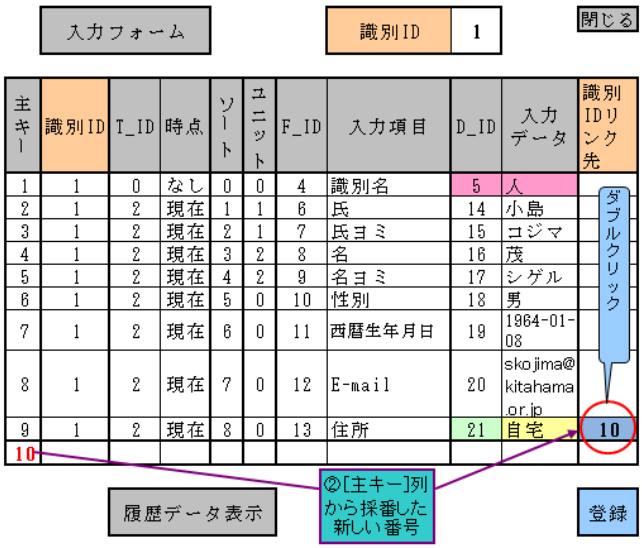

図5　リンク先の作成

5.2　入力補助機能　ーリストボックスー

自宅を識別化する入力フォームから「リストボックス機能」を使い[D_ID]と[入力データ]へデータを転記する機能を紹介する.

- 識別 ID 21 として「自宅」の入力画面が完成
- 24 都道府県名 をダブルクリックしリストボックスとして今から使用する都道府県名識別化フォームを続けて作成する（図 6）

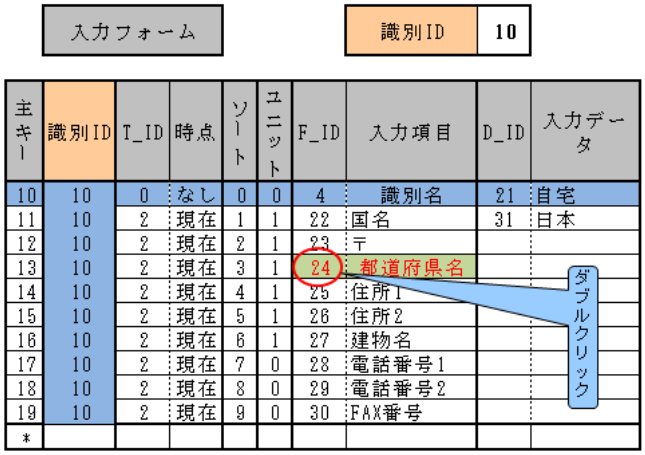

図6　自宅の入力

- 入力フォームは，テーブル「Identify_Entity_tbl」から[T_ID]= 0 AND [D_ID] =24 という検索条件を満たすレコードを探し存在していればその[識別 ID]を選択する画面を提供．存在してなければ[識別名][24][都道府県]という[識別 ID] を新規作成する．
  データとして [34][大阪府][35][東京都]をそれぞれ登録する（図 7）．

注）図 6，図 7 で画面切り替えのイベントとして「ダブルクリック」を使用すると書いたが，実際には，[主キー]をダブルクリックしたら，レコードの挿入または削除なのかを質問するメッセージ BOX を出すことになるであろう．実際に画面を構築されるプログラム作成者に画面切り替え機能は委ねることとする．

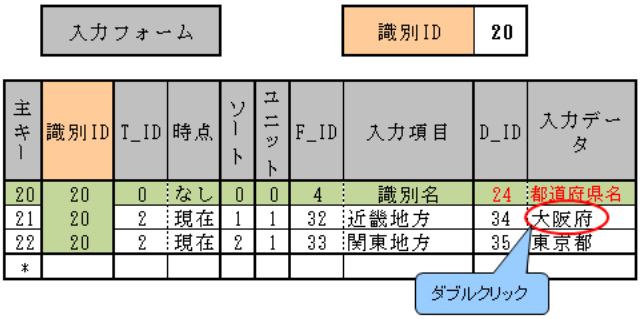

図7　都道府県名の入力

- 図 7 で，[34][大阪府]をダブルクリックし，[識別ID][21][自宅]の入力画面に戻り [24][都道府県名]の[入力データ]へ[34][大阪府]の転記を完了することになる．

5.3　入力補助機能　ー利用回数を学習するー

入力に良く使われる順から都道府県名を表示させたい場合．事前に[ソート]へ桁数の多い値を設定しておき，リストボックスとして利用され選択される度に入力フォームが，現在の[ソート]の値から，1 を引いた数字に更新していけば利用回数が多い順に表示されるはずであるが最初から[ソート]キーを降順にしておく，フォーム上の指定によりストアドプロシージャへソートの昇降を指定するのも手であろう．

5.4　入力補助機能　ー登録データの検索と利用ー

図8は，現在登録されている全[入力項目]に対して，「男」というキーワードで検索した結果[D_ID]に[18]が表示された．

この方法ならリストボックス用の[識別ID]をわざわざ用意しなくても良い．

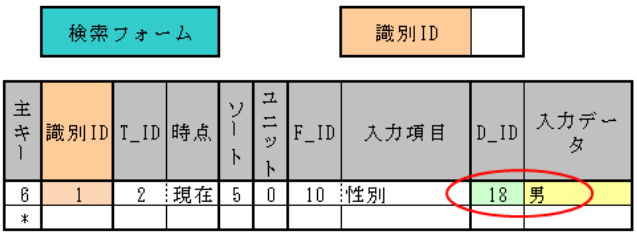

図8　登録データの検索と利用

このニューロ型データベースモデリングの特徴は，[識別ID]の登録が完了したと同時に[言葉]列のキーワード検索が可能になっている．例えば，日付（1964-01-08）や備考（独立フォームなので）にも，[主キー]を割り当てるので，テーブル「Word Entity_tbl」の[言葉]列を検索し該当が無い場合は「データベース上にデータは存在しない」ということになる．

もしも，該当レコードが見つかれば，該当した[主キー]が登録されている[識別 ID]だけを抽出すれば良い．これは日々増加して行くテーブル「Identify Entity_tbl」に対して毎回全件検索をする無駄を軽減させ，且つ非常に高速な検索が可能である．

5.5「関係」を記憶する －識別 ID リンク先を作成－

図9は，新たに「人」を追加し，それを「実弟」として人間関係を登録する．

- ［主キー］： 23 （自動取得）
- ［識別ID］： 1 （自動登録）
- ［T_ID］： 2 （自動登録）
- ［[時点］： 現在 （自動表示）
- ［ソート］： 9 （自動連番）
- ［ユニット］： 0 （自動登録）
- ［F_ID］： 36 （言葉登録処理後自動表示）
- ［入力項目］： 実弟
- ［D_ID］： 5 （言葉登録処理後自動表示）
- [入力データ]：人
- ［識別IDリンク先］：空欄をダブルクリック
  IDを指定しないので，［識別名］[5]［人］を使い、
  [D_ID][入力データ][識別IDリンク先]以外の
  項目を複製し[識別ID][24]を自動作成する
- ［識別 ID リンク先］：識別 ID 番号（主キー番号）
  24 （自動登録）

入力フォーム　識別ID 1　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
1	1	0	なし	0	0	4	識別名	5	人	
2	1	3	現在	1	1	6	氏	14	小島	
3	1	3	現在	2	1	7	氏ヨミ	15	コジマ	
4	1	2	現在	3	2	8	名	16	茂	
5	1	2	現在	4	2	9	名ヨミ	17	シゲル	
6	1	2	現在	5	0	10	性別	18	男	
7	1	2	現在	6	0	11	西暦生年月日	19	1964-01-08	
8	1	2	現在	7	0	12	E-mail	20	skojima@kitahama.or.jp	
9	1	2	現在	8	0	13	住所	21	自宅	10
23	1	2	現在	9	0	36	実弟	5	人	24
24										

図9　識別 ID リンク先を作成

図10は，[識別ID][10]の入力フォーム画面へ移動したものである。これから「実弟」の個人報を登録する．

登録した[識別ID][24]を、人間関係である「実弟」として[識別IDリンク先]に登録する

[14][小島]、[15][コジマ]は既に「言葉」に登録済なので、[D_ID]は自動表示される．

グループ番号である[識別ID]は新しく識別名を追加する度にレコード先頭の主キー値を利用するのでグループ番号（[識別ID]）は1ずつ増加するシーケンシャルな数値にはならない．

- ［D_ID］： 14 （既存データ：自動表示）
- [入力データ]：小島
- ［D_ID］： 15 （既存データ：自動表示）
- [入力データ]：コジマ
- ［D_ID］： 36 （新規データ：言葉登録処理後自動表示）
- [入力データ]：栄作
- ［D_ID］： 37 （新規データ：言葉登録処理後自動表示）
- [入力データ]：エイサク

入力フォーム　識別ID 24　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
24	24	0	なし	0	0	4	識別名	5	人	
25	24	2	現在	1	1	6	氏	14	小島	
26	24	2	現在	2	1	7	氏ヨミ	15	コジマ	
27	24	2	現在	3	2	8	名	36	栄作	
28	24	2	現在	4	2	9	名ヨミ	37	エイサク	
29	24	2	現在	5	0	10	性別			
30	24	2	現在	6	0	11	西暦生年月日			
31	24	2	現在	7	0	12	E-mail			
32	24	2	現在	8	0	13	住所			
33	24	2	現在	9	0	36	実弟			
*										

登録

図 10　[識別 ID][1]を複製し[識別 ID][24]を作成

5.6「独立フォーム」の指定－プロパテイの追加－

図11は，［識別名］[5]［人］から入力データを削除した状態で項目を複製してきた．今後，［識別名］[5]［人］に項目が追加された場合，全[識別名][5][人]に対して自動で同じ項目を追加するという機能が必要になる．その時，自動一括更新対象にしたくない[識別ID]がある場合，「独立フォーム」として区別する必要があり，「プロパテイ（属性）」を追加し，後日，自分と同じ[識別名][5][人]全レコード対象の処理が実行された時，一括処理から外してもらう等の判定情報として使用する．

今回は，［主キー］[33]の[36][実弟]を削除（履歴化）し，新たに[主キー][34] [入力項目][独立フォーム]を追加し，[T_ID][0][時点][なし]の「プロパテイ（属性）」として追加する．

- ［主キー］： 33 （自動取得）
- ［識別ID］： 24 （自動登録）
  ↓データを履歴化する為[T_ID]を[3][過去]に変更
- ［T_ID］： 3 （自動登録）
- ［[時点］： 過去 （自動表示）
- ［ソート］： 9 （自動連番）
- ［ユニット］： 0 （自動登録）
- ［F_ID］： 36 （自動表示）
- ［入力項目］： 実弟

↓「プロパテイ（属性）」を追加．

- ［主キー］： 34 （自動取得）
- ［識別ID］： 24 （自動登録）
- ［T_ID］： 0 （自動登録）
- ［[時点］： なし （自動表示）
- ［ソート］： 1 （自動連番）
- ［ユニット］： 0 （自動登録）
- ［F_ID］： 0 （登録処理後自動表示）
- ［入力項目］： 独立フォーム

入力フォーム　識別ID 24　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
24	24	0	なし	0	0	4	識別名	5	人	
34	24	0	なし	1	0	0	なし	38	独立フォーム	
37	24	1	未来	1	1	6	氏	41	大島	
38	24	1	未来	2	1	7	氏ヨミ	42	オオシマ	
25	24	2	現在	1	1	6	氏	14	小島	
26	24	2	現在	2	1	7	氏ヨミ	15	コジマ	
27	24	2	現在	3	2	8	名	36	栄作	
28	24	2	現在	4	2	9	名ヨミ	37	エイサク	
29	24	2	現在	5	0	10	性別			
30	24	2	現在	6	0	11	西暦生年月日			
31	24	2	現在	7	0	12	E-mail			
32	24	2	現在	8	0	13	住所			
33	24	3	過去	9	0	36	実弟			
*										

履歴データ表示　登録

図 11　「独立フォーム」に変更

図11は，[36][実弟]が，[T_ID][3][時点][過去]に変更された，[38][独立フォーム]が登録された．これにより，今後の入力フォームには「履歴（過去）データ表示」ボタンが必要になる．勿論，使用者の好みで，現在，過去，未来，全てをデフォルトでは表示することにしている場合は不要である．

注）［主キー］: 37 ， 38 にある[1][未来]は5.8で解説

5.7　入力データの変更（履歴化）－ユニットの役割－

人は自分の「氏」や「名」を変更することがある．入力データの変更処理は，レコード内の[D_ID]を新しい ID 番号で上書き修正はせず履歴化して残しておく．

まず[6][氏]と同じ[ユニット]番[1]である[7][氏ヨミ]のデータも[3][過去] に変更する．

最後に，履歴化された同じ「ユニット」番号のレコードから[D_ID]と[入力データ]を削除した新しいレコードを[2][現在]として追加する．

実際のプログラム処理は，履歴化する前のレコードから[D_ID]と[入力データ]を削除した物を新たに追加することになるであろう（図 12）．

【履歴化処理】

- [D_ID][14]をダブルクリック
  「履歴化するか？」という質問に「はい」を選択
- ［主キー］: 2 ， 3
- ［識別 ID］:（変更無し）
- ［T_ID］: 3 （自動変更）
- ［時点］: 過去 （自動変更）
- ［ソート］:（変更無し）
- ［ユニット］:（変更無し）
- ［F_ID］:（変更無し）
- [入力項目]:（変更無し）
- ［D_ID］:（変更無し）
- [入力データ]:（変更無し）

【新しく登録するデータ処理】

- ［主キー］: 35 ， 36 （自動取得）
- ［識別 ID］: 1 （複製元と同じ値を継承）
- ［T_ID］: 2 （自動登録）
- ［時点］: 現在 （自動表示）
- ［ソート］: 1 ， 2 （複製元と同じ値を継承）
- ［ユニット］: 1 （複製元と同じ値を継承）
- ［F_ID］: 6 ， 7 （複製元と同じ値を継承）
- [入力項目]: 氏 ， 氏ヨミ（複製元と同じ値を継承）
- ［D_ID］: 39 ， 40 （自動表示）
- [入力データ]: 中島 ，ナカジマ（手入力）

入力フォーム　識別ID 1　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
1	1	0	なし	0	0	4	識別名	5	人	
35	1	2	現在	1	1	6	氏	39	中島	
36	1	2	現在	2	1	7	氏ヨミ	40	ナカジマ	
4	1	2	現在	3	2	8	名	16	茂	
5	1	2	現在	4	2	9	名ヨミ	17	シゲル	
6	1	2	現在	5	0	10	性別	18	男	
7	1	2	現在	6	0	11	西暦生年月日	19	1964-01-08	
8	1	2	現在	7	0	12	E-mail	20	skojima@kitahama.or.jp	
9	1	2	現在	8	0	13	住所	21	自宅	10
23	1	2	現在	9	0	36	実弟	5	人	24
2	1	3	過去	1	1	6	氏	14	小島	
3	1	3	過去	2	1	7	氏ヨミ	15	コジマ	
*										

ダブルクリック後に「履歴化」され

履歴データ表示　登録

図 12　ユニットの役割

この「ユニット」は関連する[入力項目]を管理する便利な機能である．

5.8　切り替え処理の予定　－タスクとの連携－

新しい「氏」の適用開始日が後日または休日である場合[開始日時]と[終了日時]を入力し登録ボタンを押せば指定日時に切り替え処理をサーバのタスクが代わりに実行する．

[開始日時]と[終了日時]には以下の値を設定する（図 13）．

- 2 現在 14 小島　　[終了日時]: 2014-7-1 0:00
- 2 現在 15 コジマ　　[終了日時]: 2014-7-1 0:00
- 1 未来 39 中島　　[開始日時]: 2014-7-1 0:00
- 1 未来 40 ナカジマ　　[開始日時]: 2014-7-1 0:00

サーバ側のタスク処理を毎時０分に実行し，[終了時間]，[開始時間]をチェックする．[未来]は→[現在]に，そして[現在]は→[過去]にそれぞれ変更される．

識別ID 1　閉じる

F_ID	入力項目	D_ID	入力データ	識別IDリンク先	開始日時	終了日時	登録者識別ID
4	識別名	5	人				
6	氏	39	中島		2014/7/1 0:00		1
7	氏ヨミ	40	ナカジマ		2014/7/1 0:00		1
6	氏	14	小島			2014/7/1 0:00	1
7	氏ヨミ	15	コジマ			2014/7/1 0:00	1
8	名	16	茂				
9	名ヨミ	17	シゲル				
10	性別	18	男				
11	西暦生年月日	19	1964-01-08				
12	E-mail	20	skojima@kitahama.or.jp				
13	住所	21	自宅	10			
36	実弟	5	人	24			

図 13　開始日時と終了日時の設定

図 14 はサーバ側でタスクが実行されるまで [1][未来]が[2][現在]よりも上部に表示されている．これによりタスク処理が予定されている事が明示され[開始日時]を参照すればいつから適用されるかが確認出来きるし，もちろん仕様としては取り消しも可能とする．

未来データの[ユニット]番号が 1 のままだと履歴化したデータも対象になる可能性があるので，ユニット番号を新たに[3]として登録する．

入力フォーム　識別ID 1　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
1	1	0	なし	0	0	4	識別名	5	人	
35	1	1	未来	1	3	6	氏	39	中島	
36	1	1	未来	2	3	7	氏ヨミ	40	ナカジマ	
2	1	2	現在	1	1	6	氏	14	小島	
3	1	2	現在	2	1	7	氏ヨミ	15	コジマ	
4	1	2	現在	3	2	8	名	16	茂	
5	1	2	現在	4	2	9	名ヨミ	17	シゲル	
6	1	2	現在	5	0	10	性別	18	男	
7	1	2	現在	6	0	11	西暦生年月日	19	1964-01-08	
8	1	2	現在	7	0	12	E-mail	20	skojima@kitahama.or.jp	
9	1	2	現在	8	0	13	住所	21	自宅	10
23	1	2	現在	9	0	36	実弟	5	人	24
*										

履歴データ表示　ダブルクリック後に「未来」登録された　登録

図 14　タスク実行前

5.9　入力データ削除処理の例外

レコードの削除は[主キー]をダブルクリックする．現レコードを完全削除するのではなく[T_ID]を 3 過去 に変更し履歴化する．後日「削除したレコードを復活させたい」「削除した人物を知りたい」等を想定している．

[識別 ID]で使用されている[主キー]をダブルクリックした場合は，[識別 ID]でグループ化された全レコードが削除（履歴化）される．

例外として完全削除が必要な場合もある．

●[入力データ]が空白のレコード
●[入力項目]の設定ミス
●追加作業途中の取り消し

については，テーブル「Identify_Entity_tbl」から該当するレコードの完全削除を認める．

6.「管理」を始める　ーテンプレートー

良く使う入力フォームは，「テンプレート（雛形）」としてリストアップしておきたい．その場合，図 15 の様に，[識別名][43][入力テンプレート]を作成してみた．

入力フォーム　識別ID 39　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	リンク先
39	39	0	なし	0	0	4	識別名	43	入力テンプレート	
42	39	2	現在	1	0	5	人			1
40	39	3	過去	1	1	5	人			24
41	39	3	過去	2	1	44	備考	45	独立フォームなので	

図 15　テンプレート

7.「検索」を始める

テーブル「Identify_Entity_tbl」で，1000 個の商品を管理する場合，**「商品 1 個につき 1 レコードのデータを登録する」**というのはいかがだろうか，商品を 1 個単位で処理することが可能になり，在庫管理で出荷（減算処理）や返品（加算処理）の UPDATE 処理をしなくても SELECT 文で在庫計算が可能になる．

他にも，例えばコンサートチケットが 2 枚あり，1 枚目のチケットは財布の中にあり，もう 1 枚は自宅の机上にあるとする．人が財布に入れて持ち歩いているチケットは，[識別名][人]である[識別 ID][1]に追加し，自宅に置いてあるチケットは[識別名][21][自宅]の[識別 ID][10]に追加する．

登録先に迷ったら，「現状(存在場所)を登録する」．

[識別名][21][自宅]に属していたことで不都合が生じた場合は，全レコードのテーブルレイアウトは同一なので，チケットの[識別 ID]を新しく所属させる[識別 ID]に書き換えればレコード移動が簡単に完了する．

[識別 ID] である[24]を，[識別 ID リンク先]の全レコードに対して検索をかければ，[識別名][1]さんの[実弟]だということが判明する（図 14）．

例えば，市町村が合併した場合以前の町村名を履歴化しておけば，旧町村名からでも検索することが可能だ．

例えば，図 14 では，実弟は[識別 ID][24]であるが，今後[識別 ID][24]に，[入力項目][実弟]が登録されたなら，[識別

ID][1]には未入力のもう一人の[実弟]がいると判るはずだ.

8.「相互利用」を始める

8.1 相手データを取り込む

このニューロ型データベースモデリングで作成されたデータなら，お互いのテーブル「Word_Entity_tbl」の[言葉]列を比較し，異なっている[ID]列を共通の ID（番号）に書き換えた後，相手にだけ存在する「言葉」は新たに自分のテーブル「Word_Entity_tbl」へ追加する．その後テーブル「Identify_Entity_tbl」内の ID（番号）で，変更があった ID（番号）を新しい ID に置換すれば，直ぐに相手のデータが参照可能になる．

相手のテーブル「Identify_Entity_tbl」を自分のデータとして結合する場合は，読み込み時に新しい[主キー]が割り当てられるので，それに合わせて相手の[識別 ID]を書き換え，その[識別 ID リンク先]を書き換えれば利用が可能になる．

[識別 ID]の結合は，どちらか片方の[識別 ID]に書き換えてしまえば良い．

[ID]を取り込む為，新たに[WK_ID]列を追加する．相手のテーブル「Word_Entity_tbl」が貰えない場合も想定し，今後，グローバルテーブル「Word_Entity_tbl」が提供される場合を考えて[G_ID]列も追加する（表 3）

まず，相手からテーブル「Word_Entity_tbl」の提供を受けられる場合，相手[ID]を[WK_ID]に登録するが，その時自分と同じ「言葉」があれば同じ「言葉」行の[WK_ID]へ相手[ID]を登録する．知らない「言葉」は新レコードとして登録する．

例：相手の[891]［女］を→[46][女][891]として新規追加

表 3　Word Entity

ID	言葉	WK_ID	G_ID
0	なし	0	0
1	未来	1	1
2	現在	2	2
3	過去	3	3
4	識別名	4	4
5	人	5	5
6	氏	6	6
7	氏ヨミ	7	7
8	名	8	8
9	名ヨミ	9	9
10	性別	10	10
11	西暦生年月日	11	11
12	E-mail	12	12
13	住所	13	13
14	小島	528	789
15	コジマ	541	793
	⟆		
46	女	891	852

表 4 は，相手からテーブル「Identify_Entity_tbl」を受け入れた状態である．

相手：[識別 ID]　→　[WK_識別 ID]
相手：[F_ID]　→　[WK_F_ID]
相手：[D_ID]　→　[WK_D_ID]
相手：[識別 ID リンク先]　→　[WK_識別 ID リンク先]

仮読込が完了したら，新たに作成された[識別 ID]の値を[識別 ID リンク先]へ転記する．この時[WK_識別 ID]と[WK_識別 ID リンク先]を使う．

この仮読込のままなら，参照フォームで，自分のデータか，一時的に取り込んだ相手のデータかを切り替えて表示確認する．

表 4　Identify Entity（相手データ取り込み後）

主キー	識別ID	WK_識別ID	T_ID	ソース	ユニット	F_ID	WK_F_ID	D_ID	WK_D_ID	識別IDリンク先	WK_識別IDリンク先
90	90	1	0	0	0		4		5		
91	90	1	2	1	1		6		528		
92	90	1	2	2	1		7		541		
					⟆						
98	90	1	2	8	0		13		21	99	10
99	99	10	2	2	1		7		15		

確認が終了したら，取り込んだデータの全部または一部を選択し，取り込みボタンを押す．

自分のテーブル「Identify_Entity_tbl」内の以下のデータを書き換え処理をする．

[WK_F_ID]　→　[ID]
[WK_D_ID]　→　[ID]

表 5 は，データを完全に取り込めた状態である．

表 5　Identify Entity（完全取り込み後）

主キー	識別ID	WK_識別ID	T_ID	ソース	ユニット	F_ID	WK_F_ID	D_ID	WK_D_ID	識別IDリンク先	WK_識別IDリンク先
90	90		0	0	0	4		5			
91	90		2	1	1	6		14			
92	90		2	2	1	7		15			
					⟆						
98	90		2	8	0	13		21		99	
99	99		2	2	1	7		15			

8.2 自分のデータを送る

組織内共通のグローバルテーブル（ニュートラルな存在）「Word_Entity_tbl」が存在している場合，各端末で作成したテーブル「Identify_Entity_tbl」内の[G_ID]に，グローバルテーブル「Word_Entity_tbl」の[ID]を登録すれば良い．データが各端末に分散している（オフライン）なら，定期的

にグローバルテーブル「Word_Entity_tbl」の[ID]と，自分のテーブル「Word_Entity_tbl」の「言葉」を比較し，テーブル「Word_Entity_tbl」の[G_ID]を最新に更新（同期）する．

組織内の別端末に該当データを送る準備として，[WK_F_ID]と[WK_D_ID]には，同期した[G_ID]が登録される（表 6）．

表 6　Identify Entity

（送信前に［G_ID］を取得しておく ）

主キー	識別ID	WK_識別ID	T_ID	ソート	ユニット	F_ID	WK_F_ID	D_ID	WK_D_ID	識別IDリンク先	WK_識別IDリンク先
90	90		0	0	0	4	4	5	5		
91	90		2	1	1	6	6	14	789		
92	90		2	2	1	7	7	15	793		
⟆											
98	90		2	8	0	13	13	21	21	99	
99	99		2	2	1	7	7	15	15		

8.3　書き出し方法（グローバル ID を利用）

送付用データを作成する時は，送信前にグローバルテーブルと同期した[G_ID]を，[WK_F_ID]と[WK_D_ID]へ転記し，ローカルデータである[F_ID]と[D_ID]のデータを削除する（表 7）．

表 7　Identify Entity（送付用データ）

主キー	識別ID	WK_識別ID	T_ID	ソート	ユニット	F_ID	WK_F_ID	D_ID	WK_D_ID	識別IDリンク先	WK_識別IDリンク先
90	90		0	0	0		4		5		
91	90		2	1	1		6		789		
92	90		2	2	1		7		793		
⟆											
98	90		2	8	0		13		21	99	
99	99		2	2	1		7		15		

8.4　データの共有と同期（[G_識別 ID]の利用）

組織内グローバルテーブル「Identify_Entity_tbl」に，ローカル側の[識別 ID]が追加され，グローバルテーブル「Identify_Entity_tbl」に新たに追加された[識別 ID]を取得した状態である．分かりやすくする為，各列名の[WK_]は[G_]に変更した．（表 8）．

今後，ローカルテーブル「Identify_Entity_tbl」の[識別 ID]郡に対して，項目やデータの追加・変更・削除等を行った場合，データの同期先である[G_識別 ID]のグローバルテーブル「Identify_Entity_tbl」へローカルの[識別 ID]郡を送信しデータの同期を行う．

表 8　Identify Entity

（ グローバルな[G_識別 ID]取得後 ）

主キー	識別ID	G_識別ID	T_ID	ソート	ユニット	F_ID	G_F_ID	D_ID	G_D_ID	識別IDリンク先	G_識別IDリンク先
90	90		0	0	0	4	4	5	5		
91	90		2	1	1	6	6	14	789		
92	90		2	2	1	7	7	15	793		
⟆											
98	90		2	8	0	13	13	21	21	99	70
99	99	70	2	2	1	7	7	15	15		

9．まとめ

現在，このニューロ型データベースモデリングに近い，縦型テーブル設計で，約 260 万個のファイルを対象にした「ファイル名検索システム」を XAMPP（MySQL と Apache）で構築し稼働させている．収集したファイル名のフルパスを縦型データベースのフォーマットに変換（ ¥ マーク毎に区切りそれぞれをレコードとして追加）して利用しているのだ．稼働しているのはサーバ専用マシーンでは無く，市販パソコンで動いているので検索には多少時間がかかる．今後，縦型データベースが定着すれば，ハードウエアの仕様も変化し検索スピードも向上すると思う．このニューロ型データベースモデリングは

1．登録した全データが検索可能
2．全ての関連情報を抽出可能
3．最初の検索で該当データの有無判定が非常に速い
4．登録データは ID 化した数値なのでコンパクト設計
5．検索プログラムの変更不要（メンテナンスフリー）
6．各データの追加・並び替え・移動・リンク・グループ化・削除（履歴化）が簡単
7．同じモデリングデータ同士なら「相互利用」可能

である．

一般的に使用されている殆どの表計算ソフトやデータベースのテーブルレイアウトは，「列」が「項目」であり，「行」は「データ」として処理されている．このニューロ型データベースモデリングは，列名とデータ行をそれぞれ縦長な列データとして連続に繋げているだけなので，図 13 で使用している「タスク処理」用の[開始日時][終了日時][登録者識別

ID]も各[識別 ID]に登録しても良い.

例えば，所属出来る ML が沢山あり，頻繁に追加や削除を行う場合各個人の[識別 ID]に，[入力項目][所属 ML]　[入力データ][DB 勉強会]．さらに，[識別 ID リンク先]に，[識別名][DB 勉強会]を作成する.

続いて次行には，[入力項目][入会年月日]　[入力データ][2014-6-30]とし，[ユニット]は，現在の最高値に 1 を加えた番号で登録する.

新しい未知の ML 名が登場する度に，上記の作業を繰り返し ML からの脱会は，各個人の[識別 ID]から該当 ML 名を見つけ，履歴化すれば良い．その時，同じ[ユニット]も同じく履歴化してしまう．必要なら最初から[入力項目][退会年月日]も追加しておく，その時[入力データ]には[0][なし]を登録しておくことにより「**5.9　入力データ削除処理の例外**　」で書いた通り，[入力データ]が空白のレコードは削除される可能性がある為この場合削除されず，未来に入力予定だと予想出来る．取りあえず所属 ML 名が分かれば良いのならこれで登録が終了する.

この後 ML に関する情報（送信先アドレス等）が必要な場合，新たに[識別 ID]を作成し，それを[識別 ID リンク先]として登録すれば良い．（図 16）.

入力フォーム　　識別ID　1　　閉じる

主キー	識別ID	T_ID	時点	ソート	ユニット	F_ID	入力項目	D_ID	入力データ	識別IDリンク先
1	1	0	なし	0	0	4	識別名	5	人	
35	1	1	未来	1	3	6	氏	39	中島	
36	1	1	未来	2	3	7	氏ヨミ	40	ナカジマ	
	1	1	未来	3	3	46	開始日時	47	2014/7/1	
	1	1	未来	4	3	48	登録者識別ID	5	人	1
2	1	2	現在	1	1	6	氏	14	小島	
3	1	2	現在	2	1	7	氏ヨミ	15	コジマ	
	1	2	現在	3	1	49	終了日時	47	2014/7/1	
	1	2	現在	4	1	48	登録者識別ID	5	人	1
4	1	2	現在	5	2	8	名	16	茂	
5	1	2	現在	6	2	9	名ヨミ	17	シゲル	
6	1	2	現在	7	0	10	性別	18	男	
7	1	2	現在	8	0	11	西暦生年月日	19	1964-01-08	
8	1	2	現在	9	0	12	E-mail	20	skojima@kitahama.or.jp	
9	1	2	現在	10	0	13	住所	21	自宅	10
23	1	2	現在	11	0	36	実弟	5	人	24
43	1	2	現在	12	4	50	所属ML	51	DB研究会	
44	1	2	現在	13	4	52	入会年月日	53	2014/6/30	
45	1	2	現在	14	4	54	退会年月日	0	なし	
*										

履歴データ表示　　[入力データ]は空白では無いので 5.9の理由によりこの行削除は不可　　登録

図 16　「タスク処理」と「他組織との関係」登録

実際にニューロ型データベースモデリングが完成してみると，各自が持っているデータを簡単に「相互利用」することが可能なテーブル構造だということに気が付いた．データベースとして利用する以外に，異機種・異 OS 間でのデータ受け渡しをするツールとしての利用も可能だ.

各組織で，①オフィシャルなグローバル「Word_Entity_tbl」を管理するサーバを立ち上げ，組織内共通の[G_ID]を提供する必要がある．その時「言葉」を MD5 等の「ハッシュ関数値」としても記憶させる為，もう 1 列データを追加すると便利かもしれない．②組織内専用で利用するテーブル「Identify_Entity_tbl」サーバを準備し，各自が管理しているデータを登録後，共有・同期して行けば組織内のデータは 1 つに集約出来る.

この入力・検索フォームを作成する場合，プログラミング言語と，SQLite 等のリレーショナルデータベース管理システム(RDBMS)が必要になる．どのプログラミング言語で作成していけば良いのかは，稼働する OS と端末の条件で決まってくると思う.

以上

[文献]

[1] 朝井淳：“SQL ポケットリファレンス”，技術評論社(1999).

[2] 堀川明：“SQLServer7.0/MSDE 完全トレーニングテキスト上”，技術評論社(2000).

[3] 堀川明：“SQLServer7.0/MSDE 完全トレーニングテキスト下”，技術評論社(2000).

[4] 梅田弘之，碓井満：“SQLServer7.0徹底入門―全機能+VB6.0アプリケーション構築”，翔泳社(2000).

[5] 国吉直樹，初音玲，清水由美，イントツーワン：“SQL 逆引き大全333の極意―Oracle/SQL Server/Microsoft Jet 対応”，秀和システム(2002).

[6] 有限会社ガリバー：“AccessプログラミングTips ポケットリファレンス”，技術評論社(2004).

[7] 若杉司：“Oracle データベース運用・管理ポケットリファレンス Oracle10g/9i対応”，技術評論社(2005).

[8] 鈴木啓修：“MySQL全機能バイブル 現場で役立つAtoZ”，技術評論社(2009)

[9] テクノロジックアート：“独習UML 第4版”，翔泳社(2009)

[10] 清野克行：“作りながら基礎から学ぶPHPによるWebアプリケーション入門 XAMPP／jQuery／HTML5で作るイマドキのWebサイト”，秀和システム(2011)

小島 茂　Shigeru KOJIMA

北浜法律事務所・外国法共同事業 情報システム課 2000 - 現在に至る．Microsoft SQL サーバで業務システムを開発．
1983 PL 学園高等学校（定時制）普通科卒業後 PL 学園女子短期大学教務課に勤務し現神戸女子大学教授 小松俊朗氏から BASIC プログラムを教わる．1995 同 短期大学のシステム管理者（大阪地域大学ネットワーク(ORIONS)と UUCP 接続を構築）．2000-2001 PL 学園女子短期大学非常勤講師「OA 機器演習」.