ERモデルに基ずくデータベースの論理設計技法とツール

近藤秀文（日立製作所システム開発研究所）
小林正和（同左）
酒井博敬（日立製作所技術研修所）

1．はじめに

データベースの設計は，(1)概念設計（または情報構造設計），(2)論理設計，および(3)物理設計，の3段階に分けることができる。第1の概念設計では，データベース応用システムが必要とするデータをデータ構造の形式で表現する。本稿では，これを概念スキーマと呼ぶ。第2の論理設計では，概念スキーマをDBMS（DataBase Management System）固有のデータモデルで表現したデータ構造の形式に変換する。本稿では，これを論理構造と呼ぶ。第3の物理設計では，論理構造のディスク上における格納構造を決定する。本稿では，これを物理構造と呼ぶ。概念スキーマ記述用，論理構造記述用，および物理構造記述用のデータモデルを，本稿では，それぞれ，概念データモデル，論理データモデル，および物理データモデルという。概念データモデルとしては，ER（Entity-Relationship）モデル，データ・アブストラクション・モデルを始めとして各種のデータモデルが提案されている［1，2，3，4，5，6］。論理データモデルとしては，階層モデル，ネットワーク・モデル，およびリレーショナル・モデルがある。物理データモデルとしては，ハッシュ構造，インバーテッド構造，マルチリスト構造，VSAM構造，あるいはこれらを組合せた構造などがある。本稿では，ERモデルを用いた概念設計，およびADM（Adaptable Data Manager：日立のDBMS）の階層モデルを用いた論理設計について述べる。

以下，次の順に説明する。2章では，設計手順の概要，概念データモデルを用いずに論理データモデルを用いて設計を開始した場合の不具合，について述べる。3章では，概念設計法とそのツール，4章では，論理設計法とそのツールについて説明し，5章で結言を述べる。

2．設計手順

図1は，データベース設計手順の概要を示したものである。まず，開発計画システムのデータベースに対するデータ要求を整理し，概念設計を行なう。概念設計では，利用者の代表と設計者とが共同で作業を行ない，概念スキーマを作成する。この概念スキーマを関連部署の利用者がレビューし，必要ならば修正を行ない，概念スキーマを確立する。次に，設計者は，データ量およびアクセス・パターンのデータを使用して論理設計を行ない，概念スキーマを論理構造に変換する。設計者は，設計結果を，他のデータ

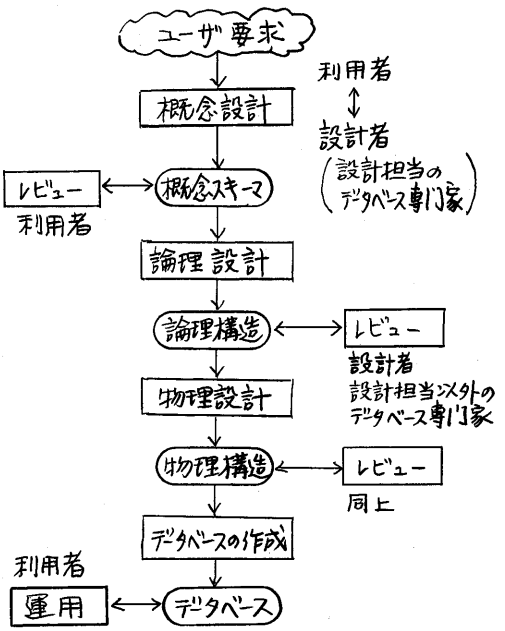


図1 データベースの設計手順

ベース専門家と協同でレビューし，論理構造を固める。最後に，同じくデータ量およびアクセス・パターンのデータを使用して物理設計を行ない，物理構造を決定する。物理設計結果のレビューを行なうことによって，すべての設計を終了する。

ここで，データベースに対するデータ要求に基ずいて，論理データモデルを用いて直接論理構造を決定する場合の欠点について述べ，概念設計の必要性を明らかにする。「CUSTOMERとOFFICEの間のCONTRACT」という内容のデータベース化要求について考える。この要求に対する論理構造として，階層モデルでは，たとえば，図2(a)，(b)，(c)，および(d)のような構造がある。図2(a)では，特定の顧客（CUSTOMER）に関する営業所（OFFICE）および契約（CONTRACT）の検索は高速に行なえるが，特定の営業所に関する顧客および契約の検索ではデータベース全体のサーチが必要になってしまう。図2(b)は，ポインタ・セグメント＊を用いて表現した論理構造である。この構造では，上記のどちらの検索も効率よく行なえる。図2(c)は，図2(a)のOFFICEをOFFICE-CODEで置換え，OFFICEを別に格納する論理構造である。この構造は，図2(a)の構造よりデータの冗長性が少なくなる。図2(d)は，CUSTOMER，OFFICE-CODE，およびCONTRACTを1つにまとめ，OFFICEを別に格納する論理構造である。この構造は，顧客が1つの営業所とだけ契約を結び，契約の記録数に制限があるという場合に便利な構造で，比較的見やすい構造であるところに特徴がある。

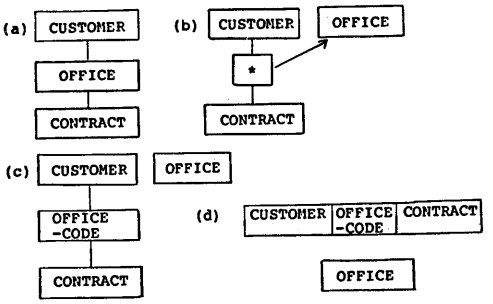


図2 階層モデルを用いた各種の論理構造

以上述べたように，同じ内容のデータを表現するのにも，論理構造では，いろいろな表現の仕方がある。これは，本来のデータの意味以外に，性能，データ冗長性，および見やすさ，の考察を同時に行なっているからである。また，データの意味に関する記述規則も充分でない。そこで，データの意味とそれ以外のものを分けて設計できることが好ましいという考えから，データの意味に関する内容は概念設計で扱うこととしたわけである。

3. 概念設計

3.1 ERモデル

概念設計で使用するERモデルの詳細は，文献[1,8,9]に記述されているが，本稿の説明のために概要を述べる。ERモデルの主な記述要素は，エンティティ，リレーションシップ，カテゴリ，アトリビュート，およびバリューである。エンティティは，個々の人，物，事象など，リレーションシップは，エンティティ間の関係，アトリビュートは，エンティティまたはリレーションシップの詳細データアイテム，バリューは，アトリビュートのデータ型および値域である。同種のエンティティの集合をEセット，同種のリレーションシップの集合をRセット，同種のバリューの集合をVセットという。カテゴリは，同種の

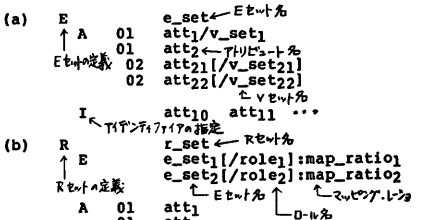


図3 Eセット，Rセット，カテゴリ，およびVセットの定義形式

Eセット間の関連を記述するためのもので，文献[2]のGeneralization の概念をERモデルに取入れたものである。ERモデルについて，図3に定義形式，図4に記述例，図5に図4の図表現を示す。

3．2　概念設計手順

概念設計は，次の3段階で実施する。(1)ローカル・ビューの作成：利用部門別にデータ要求を整理し，ERモデルで記述する。通常は，一度に開発計画システム全体の情報構造を明確にするのが困難なため，個別にデータのモデル化を行なうわけである。図4および図5の(a)，(b)，および(c)は，3つのローカル・ビューの例である。ここで，Rセットの単純化を行なっておくことが重要な条件である。たとえば，「TEACHER が CLASS で STUDENT を教える」というRセットは，もしどの先生もあるクラスに属する学生全体に教えるということであれば，「TEACHER は CLASS で教える」と「STUDENT は CLASS に属する」という2つのRセットに分割しておく。(2)ローカル・ビューの統合による概念スキーマの作成：ローカル・ビュー間では，シノニム（異名同義）およびホモニム（同名異義）の問題が発生する。このため，図4(d)のシノニム定義例のような定義を行ない，これを使用して統合を行なう。図6は，統合結果のER図である。(3)概念スキーマを対象としたデータ量およびアクセス・パターンの整理：表1は，ERモデルに対する基本アクセス・パターンを示したものである。たとえば，表1のaは，エンティティのアトリビュートをキィとした同じエンティティの他のアトリビュートの

```
E          CUSTOMER
  A  01    C_CODE
     01    C_NAME
     01    C_TEL
  I        C_CODE

E          OFFICE
  A  01    O_CODE
     01    O_NAME
     01    O_TEL
     01    O_ADDR
  I        O_CODE

R          CONTRACT
  E        CUSTOMER : N
           OFFICE : M
  A  01    CONT_HISTORY
     02    T_DATE
     02    T_SPEC
```

(a) ローカル・ビュー：CONTRACT_INF

```
E          CUSTOM
  A  01    CUSTOM_CODE
     01    C_NAME
     01    C_TEL
```

(c) ローカル・ビュー：SALES_INF

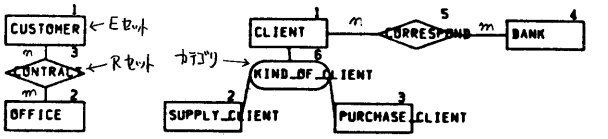
```
E          CLIENT
  A  01    CLIENT_CODE
     01    C_NAME
     01    C_TEL
  I        CLIENT_CODE

E          SUPPLY_CLIENT
  A  01    CASH_OR_CHECK

E          PURCHASE_CLIENT
  A  01    DAYS_FOR_SHIPMENT

C          KIND_OF_CLIENT
  G        CLIENT
  S        SUPPLY_CLIENT
           PURCHASE_CLIENT
```

(b) ローカル・ビュー：CLIENT_INF

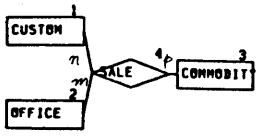
```
Y
  E    CONTRACT-INF.CUSTOMER
       SALE_INF.CUSTOM
       CLIENT_INF.SUPPLY_CLIENT
  EA   CONTRACT_INF.CUSTOMER.C_CODE
       CLIENT_INF.CLIENT.CLIENT_CODE
       SALES_INF.CUSTOM.CUSTOM_CODE
```

(d) Eセットおよびアトリビュートに関するシノニム

図4　ローカル・ビューおよびシノニムの記述例



(a) ローカル・ビュー：CONTRACT_INF　　(b) ローカル・ビュー：CLIENT_INF



(c) ローカル・ビュー：SALES_INF

図5　ERモデルを用いたローカル・ビューの記述例

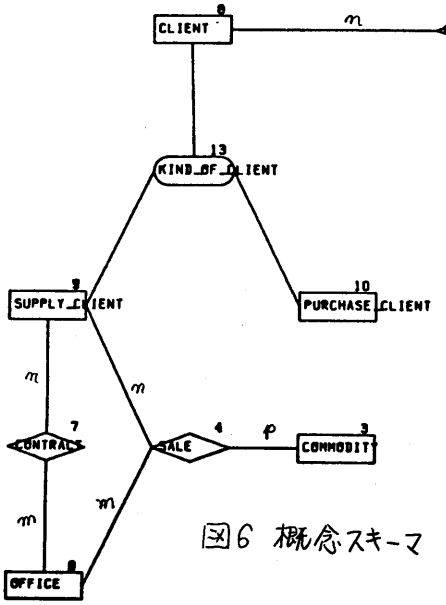


図6　概念スキーマ

検索，bは，エンティティのアトリビュートをキィとしたリレーションシップのアトリビュートの検索を示すものである。図7(a)は，基本アクセス・パターンを用いた記述例である。図7(b)は，データ量に関する記述例である。上述した概念スキーマ，データ量，およびアクセス・パターンは論理設計で使用する。

3．3　概念設計ツール

概念設計ツールDBDS/CSDA（Database Design Support system / Conceptual Schema Design Aid）は，図4のようなローカル・ビューのデータを入力すると，図5および図6のER図を出力すると共に，その内容をデータベースに保存する。図4のデータは対話形式で入力する。図8に対話コマンドの種類，図9にコマンド実行例を示す。図7に示したデータ量およびアクセス・パターンに関するデータはファイルに保存する。

4．論理設計

4．1　階層モデル

階層モデルの詳細は，文献[7]にあるが，その主要素は，図10に示すような，フィールド，セグメント，ルート・セグメント，ポインタ・セグメント，および従属セグメントである。ポインタ・セグメントは，＊または＊＊で表わす。＊は1方向，＊＊は両方向のポインタ・セグメントであることを示す。図10(a)の左または右に示す構造をPDBR（Physical Database Record）タイプという。また，ルート・セグメントのキィ・フィールド以外のフィールドを用いて直接検索を可能にする構造をセカンダリ・インデキシングという。図10(b)に示すような，PDBRタイプに従がう実データをPDBRオカレンスという。PDBRオカレンスは，ディスク上ではポインタ等を使用して，例えば，図10(c)のような形で格納される。

表1　基本的アクセス・パターン

Operation		Access pattern
GET	a	E.att->E/E.att
	b	E.att->R/R.att
	c	E.att->R->E/E.att
	d	E.att,R.att->E/E>att
	e	E.att,R.att->R/R.att
	f	R.att->E/E.att
	g	R.att-.R/R.att
	h	φ ->E/E.att
	i	φ ->R/R.att
ADD/ DELETE	j	E/E.att
	k	R/R.att
MODIFY	l	E.att
	m	R.att

```
AR   CLIENT_BANK LIST ON OFFICE
 A   1   GE   C:      OFFICE(O_CODE)          ←検索キィ指定
              T:VIA   CONTRACT                ←検索対象
              T:ALL   SUPPLY_CLIENT(C_CODE)   ←検索対象
       2  GE  T:VIA   CORRESPOND
              T:ALL   BANK
       2  GR  T:ALL   SALE
        3 GE  T:ALL   COMMODITY(COM_NAME)
              T:ALL   OFFICE(O_NAME)
 F   20/MONTH
 W   5.5          ← 重要度
```

(a)　アクセス・パターン

```
Q
 E   OFFICE                 100  ←営業所の数
  A  O_NAME            1.02/1.0
     O_TEL             1.0/3.0   ←1つの営業所に対するtelの数 ／ 1つのtelに対する営業所の数
 R   CONTRACT              1500  ←リレーションシップ(契約)の数
  E  CUSTOMER                 3  ←1つの顧客に対するリレーションシップの数
     OFFICE                 150
  A  CONT_HISTORY      1.0/2.0
```

(b)　データ量

図7　アクセス・パターンとデータ量の記述例

Objects ＼ Operations	Create	Get	Modify	Delete
Schema	CS	GS	MS	DS
local vieW	CW	GW	MW	DW
sYnonym	CY	GY	MY	DY
Homonym	CH	GH	MH	DH
E-set	CE	GE	ME	DE
R-set	CR	GR	MR	DR
Category	CC	GC	MC	DC
V-set	CV	GV	MV	DV
Attribute	CA	—	MA	DA

図8　コマンドの種類

```
Command ?
— CW  ← C(作成)・W(ローカル・ビュー)のコマンドを入力
Name ?
— EXAMPLE1  ← ローカル・ビューの名称を入力
Command ?
— CE
Name ?
— EMPLOYEE
Position of co-ordinates ?
— 65, 75  ← 画面上の位置(又は)座標の指定
[l-no]:  A [/V] Name [R (range)] ?
— name/NAME
— address/ADDR
— tel/TEL
— job-type/J-TYPE  ← Vセット / アトリビュート
— @
Identifier ?
— name
— @  ← 入力の終了
Comment ?
— @
Command ?
— CE
Name ?
— PROJECT
Position of co-ordinates ?
— 85, 75
[l-no]:  A [/V] Name [R (range)] ?
— *
Command ?
— CR
Name ?
— WORKS
Position of co-ordinates ?
— 75, 75
E Name [/Role] [: Mapp.] ?
— EMPLOYEE/WORKER : N
— PROJECT/CONTRACT-PROJECT : M
   ・
   ・
   ・
```

—で始まる行は設計者の入力を示す。

図9　コマンドの実行例

アクセス法としては，ＨＳＡＭ，ＨＩＳＡＭ，ＨＤＡＭ，およびＨＩＤＡＭの４種類がある。

４．２　設計パラメタ

階層モデルの設計パラメタは，大きく分けると，ＰＤＢＲタイプ，セカンダリ・インデキシング，ポインタ，およびアクセス法の４種類である。後の２つは，ＰＤＢＲオカレンスをディスク上に格納するための技術であるので，物理設計のパラメタとし，本稿の論理設計では，前の２つを設計の対象とする。階層構造の物理設計については，文献[10,11]がある。

４．３　論理設計手順

ＥＲモデルから階層モデルへの変換については，文献[8,10,12]がある。しかし，これらの研究では，論理構造の候補が一面的な基準で作成されており，（たとえば，ポインタ・セグメント＊で実現），設計者の選択の余地が少なかった。階層モデルを用いた論理構造作成のポイントはセグメントの作り方である。これは，基本的に，(1)Ｅセットのアイデンティファイアで構成，(2)ポインタ・セグメントで構成，および(3)ＥセットまたはＲセットのアトリビュートで構成，の３種類の方法がある。本稿では，これらの３種類の構成方法を，(a)性能，(b)データ冗長性，および(c)構造の見やすさ・好み，の３つの観点から検討できる設計手順を作成した。

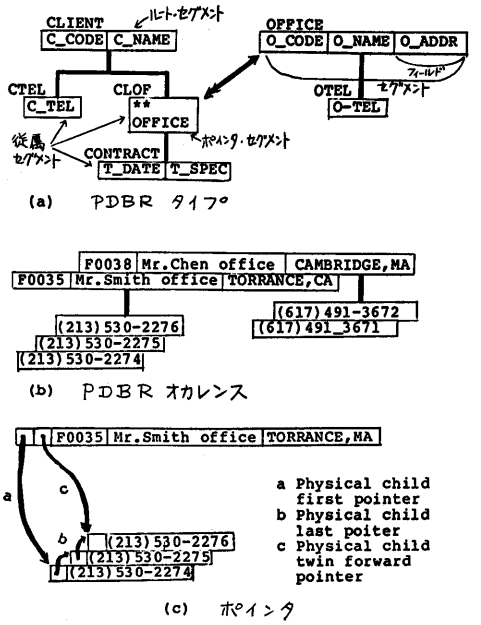


図10　PDBRタイプ，PDBRオカレンス，およびポインタ

設計手順では，次の記号を使用する。(1)セグメント階層：図10(a)の左のPDBRタイプをCLI-CONT(CLIENT:(CTEL),(**OFFICE:(CONTRACT)))のように表わす。曖昧でない限り括弧は省略できる。(2)すべてのセグメント階層の組合せ：(A:B:C),(A:C:B),…,(C:B:A)の全体を(<A,B,C>)で表わす。(3)セグメント連結：セグメントまたはフィールドA，B，Cを１つのセグメントにするとき，A＋B＋Cと表わす。(4)セグメント結合：２つのセグメント階層P，Qを結合して１つのセグメント階層にするとき，P(A:B:C)[A.PA1]*[A.QA1]Q(A:D:E)で表わす。PA1，QA1は，それぞれ，セグメントAのキィフィールドである。曖昧でないとき P*Q で表わす。(5)Ｅセット，Ｒセット，カテゴリ：ＥセットをE[A_1,…,A_m]，ＲセットをR[E_1,…,E_k:A_1,…,A_m]，カテゴリをC[E_1:E_2,…,E_p]で表わす。括弧内のE_iはＥセット，A_iはアトリビュートを表わす。カテゴリのE_1は抽象化Ｅセット，E_2,…,E_pは細分化Ｅセットを表わす。

図11は，論理設計手順を表わしたものである。ボックス１－２はカテゴリのセグメント化，ボックス３－９はＲセットのセグメント階層化，ボックス10－15はセグメント階層の統合による論理構造の作成である。

４．４　論理設計ツール

図６の概念スキーマ，図７のデータ量およびアクセス・パターンを入力とし，図11の設計手順に従って論理構造の候補を作成するツールDBDS／LSDA(Logical Structure Design Aid)を現在開発中である。

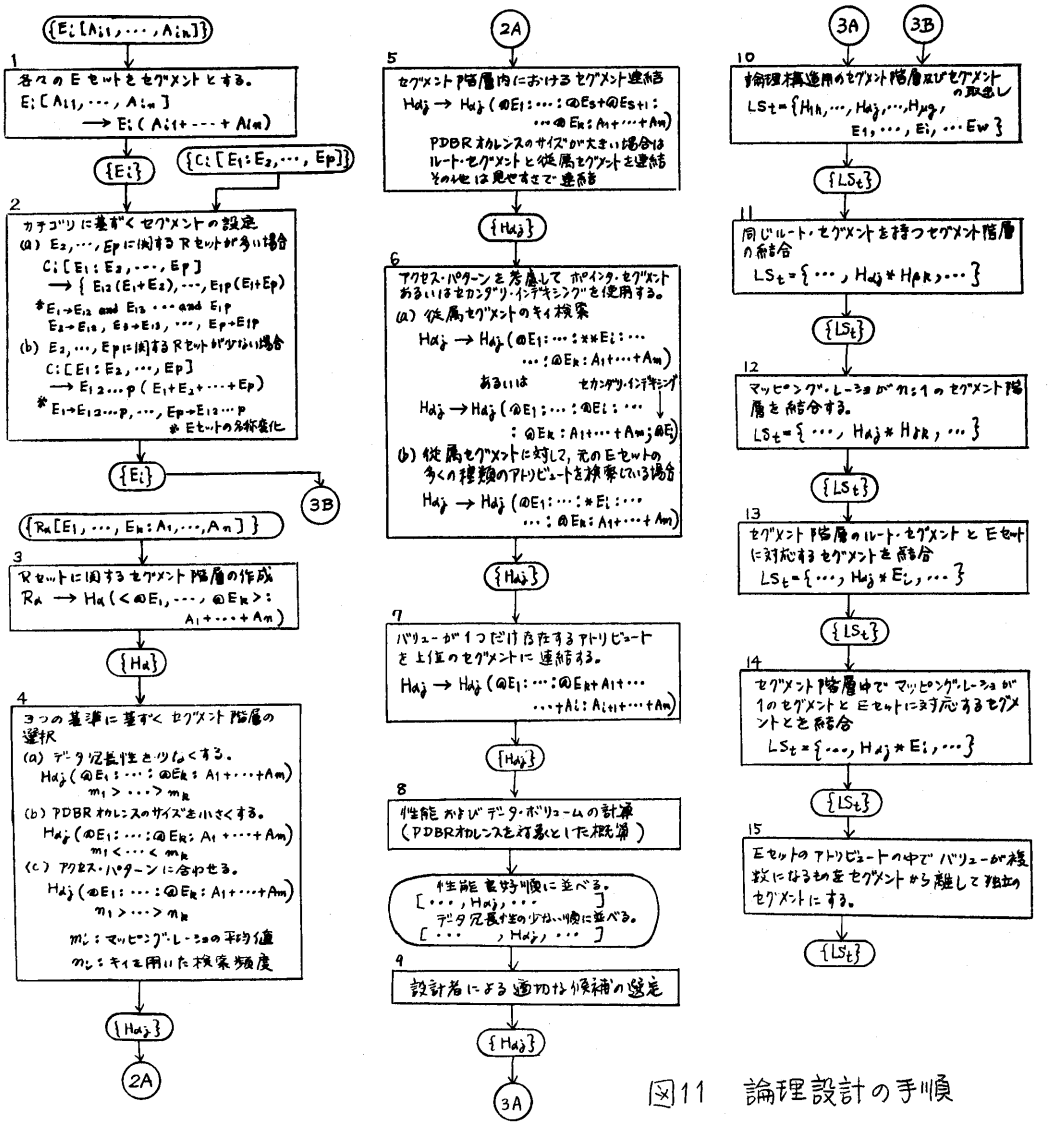


図11 論理設計の手順

5. おわりに

本設計法の適用事例では良好な結果を得ているが，システム，ソフトウェア，プログラムの各設計法との一貫した設計技法は今後の課題である。なお，本研究に御協力頂いた同社の石井信子，河崎善司郎，吉田都三，高橋美喜子，加藤孝，堀内一の各氏に感謝致します。

参考文献

[1] Chen, P., "The entity…", ACM TODS Vol.1 No.1 (1976)
[2] Smith, J.M., "Database…", ACM TODS Vol.2 No.2 (1977)
[3] 穂高, データベースの論理設計, 情報処理学会 (1981)
[4] 有沢, "Entity-Association…", 同上 Advanced DBS symp. (1981)
[5] 田中, "データベースとデータ独立化…", 北川氏の情報システム研究会 (1980)
[6] 真名垣, "A data base…", IEEE COMPSAC (1979)
[7] Date, C.J., An Introduction to …, 3rd edition (1981)
[8] 西井, "A unified …", Int. Conf. on ER approach (1979)
[9] 西井, "A method for …", Int. Conf. on VLDB (1981)
[10] Lusk, E., "A practical design …", ACM SIGMOD (1980)
[11] 近藤, "A database design …", IEEE COMPSAC (1980)
[12] Tsao, J., "Enterprise schema …", Int. Conf. on ER (1979)