

実体関連モデルDBMSの一試作

山崎 剛, 瀬川 進*, 落水 浩一郎
(静岡大学・工学部・情報工学科)

1. はじめに

われわれは、小規模のプログラミング・チームが、効率的に、ソフトウェアの開発、保守、開発成果の蓄積を行なうことのできる環境作りを目指した、計算機支援システムの開発を進めている。

その基本的考え方、および目標は、(1)ソフトウェア・ライフサイクルにおけるプログラミング・チームの活動は、各種設計情報の生成、検索、蓄積、変換、等の活動であるという観点より、それらの設計情報を、データベース化して管理すると共に、データベースのアクセスを支援するツール群を配置し、各種プログラミングの活動を支援する。(2)複数の人間による協同作業を円滑にするための、設計情報の共有、保護、分配、交信、蓄積、等のメカニズムを明らかにする。という点にある^{(1),(2)}。

設計情報データベースを中心として、プログラミング開発、保守支援を行なう立場においては、(1)適切なプログラム設計情報構造の認識、(2)データベース・アクセスに関して、プログラミング支援に必要な機能、性能を、十分に提供するデータベース管理システムの存在は、必須である。この2点に関して、以下のような成果を得た。

(1)データモデルとしては、実体関連モデル⁽³⁾を採用し、われわれの所属する研究室が、日常行なっているプログラミング活動の、各ライフサイクル局面で取り扱われる設計情報構造を、認識した。実体関連モデルを採用した理由は、実体と実体間の関連を分離して考える、というような、実体関連モデル自体のもつ性質が、ライフサイクルの各局面において生成、利用される各種設計情報の基本単位、基本単位間の関係を定める、という、現実世界のデータ構造の認識に、素直に適用できたことと、対応する物理構造の設計において、①性能を考慮したデータの格納法、②ハッシングによる高速アクセスの手段、③主キーの更新を許し、存在従属性に起因する処理を、効率的に行なうための手法、等のデータベース構築技法を、定め得ることができたことである。

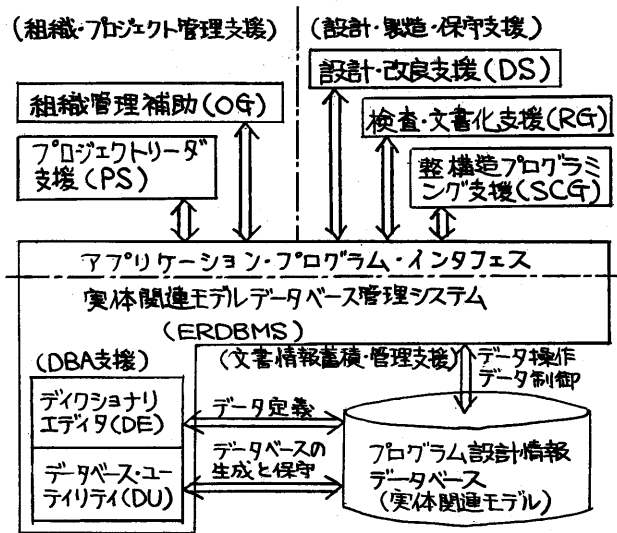
(2)ホストマシン(FACOM 230-45S)とのトレードを考慮に入れつつ、次に示す各機能を提供する実体関連モデルデータベース管理システムを実現した。①データ操作機能は、関係言語族の分類によれば、要素操作型に相当し、現在、検討中である非手続き的言語を、上重ねることを目的としている。②データ制御は、本システムが、プログラミング支援専用であるという使用環境を考慮に入れ、利用者、アクセス対象、データ操作によって定まる呼出し制御を行なう。また、同時実行制御も行なう。③データ定義は、概念スキーマと内部スキーマを、別々に定義でき、端末のデータベース管理者が、簡単なデータ定義言語を、対話形式で入力することにより行なう。④利用統計の記録、性能の測定を、自動的に行なう。なお、プログラム設計情報構造は、二項関連で十分に表現できるため、二項関連のみを扱うシステムとして実現したが、容易に、多項関連に拡張可能ようにしてある。

*現在、松下電器(株)。

2. システムの環境

実体関連モデルデータベース管理システム（以後略してERDBMSと言う。）の環境を、図1に示す。

ERDBMSは、アプリケーション・プログラム・インタフェースを通して、各アプリケーション・プログラム（各種プログラミング支援ツール）に対し、データ操作、制御機能を提供する。また、ERDBMSは、データベース管理（DBA）支援サブシステムとして、ディクショナリ・エディタ（DE）と、データベース・ユーティリティ（DU）をもち、それぞれ、データ定義機能、システムの運営を円滑にするための支援機能群を、データベース管理者に対して提供する。



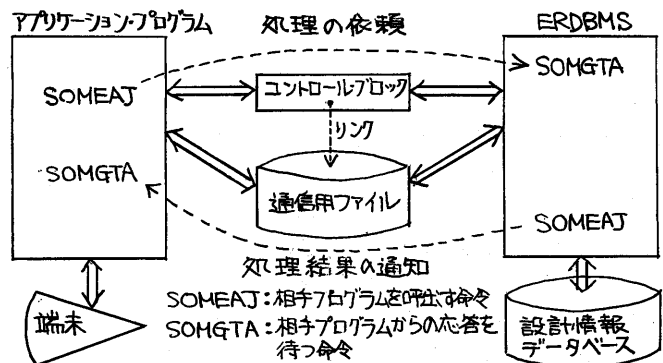
〔図1〕 システムの環境

3. データ操作機能

ERDBMSと各種アプリケーション・プログラムは、独立したタスクとして構成され、互いに相手のプログラムを呼び出して、データの受取を行なう。交信の際のパラメータやデータの受け渡し方法は、ADABAS⁽⁴⁾に準じたもので、通信制御用のコントロール・ブロックと、それにリンクされた通信用ファイルを用いて行なう（図2）。

コントロール・ブロックの構造を図3に、ERDBMSが提供するデータ操作機能を図4に示す。これらの図において、オカーレンスとは、個々のentityやrelationshipを示すものである。

アプリケーション



〔図2〕 ERDBMSとアプリケーション・プログラムの交信

DCL 1	CONTROLL_BLOCK,	
2	APPLICATION_PROGRAM_NAME	CHAR(4), アプリケーション・プログラム名
2	FUNCTION	CHAR(2), データ操作機能
2	PARAMETER	BIN FIXED, パラメータ
2	PASS_MODE	BIN FIXED, 経路の選択
2	SET_NUMBER=1	BIN FIXED, }対象とするset
2	SET_NUMBER=2	BIN FIXED, }
2	OPTION	BIN FIXED,
2	NUMBER_OF_ATTRIBUTE	BIN FIXED, 対象とするアトリビュートの数
2	NUMBER_OF_OCCURRENCE	BIN FIXED, 対象とするオカーレンスの数
2	END_STATUS(2)	BIT(16), 処理結果
2	BUFFER_LENGTH(5)	BIN FIXED,
2	NUMBER_OF_BLOCK	BIN FIXED, }通信用ファイルに関する
2	FILE_NAME	CHAR(26), 情報
2	VOLUME_NAME	CHAR(6);

〔図3〕 コントロール・ブロックの構造

ン・プログラムは、コントロール・プログラムの所定の項目をセットし、ERDBMSに処理を依頼する。

ERDBMSは、その処理結果を、END-STATUSにセットし、アプリケーション・プログラムに通知する。

	データ操作機能		説明
(1)	OPEN	DBMS	DBMSとアプリケーション・プログラムを結合する。
(2)	CLOSE	DBMS	DBMSとアプリケーション・プログラムを切り離す。
(3)	SET	PASS	存在従属するentityを指定する。
(4)	RETRIEVE	OCCURRENCE	オカレンスを検索する。
(5)	ADD	OCCURRENCE	オカレンスを登録する。
(6)	DELETE	OCCURRENCE	オカレンスを削除する。
(7)	GET	ATTRIBUTE	アトリビュートの値を読み出す。
(8)	UPDATE	ATTRIBUTE	アトリビュートの値を更新する。

〔図4〕 データ操作機能

登録、削除するオカ

レンス、検索条件、読み出すアトリビュートの値の出力形式、アトリビュートの値の更新内容、等は、通信用ファイルを用いて指定する。また、検索されたオカレンスや、読み出されたアトリビュートの値も、通信用ファイルに格納される。紙面の都合上、詳細な指定法については、省略する。

以下、個々のデータ操作機能について、システム内の処理も含めて説明する。

(1)、(2)は、ERDBMSの利用開始、終了のためのもので、システムは、アプリケーション・プログラム名を登録し、データベース利用者のアクセス範囲の設定を行なう。アクセス範囲については、4章を参照。

(3)は、存在従属性のあるオカレンス(weak entity/relationship)を指定するときに、そのオカレンスが従属するentityを指定するためのものである。

(4)は、オカレンスを検索するためのもので、アトリビュートの値の条件や、あるentityに一段のrelationshipでつながっている、というような、entity間の関連に関する条件をもとに、検索できる。検索条件として、いくつかの条件に、論理和、論理積、および否定の演算を施すことが可能である。

(5)、(6)は、オカレンスを、登録、削除するためのものである。regular entity/relationshipの登録は、単独に行なうが、weak entityの登録は、weak relationshipと対で行なう。entityを削除すると、そのentityに関連するすべてのrelationshipと、そのentityに存在従属であるすべてのentity、および、relationshipが、自動的に削除される。regular relationshipは、単独に削除できるが、weak relationshipの単独削除は許されない。これらの制限と処理は、データベース内に、孤立したentityや、無意味なrelationshipが生じないように考慮した結果である。

(7)、(8)は、アトリビュートの値の読み出し、更新を行なうためのものである。entityの主キーの更新は許されるが、relationshipの主キーの更新は許さない。その理由は、relationshipの主キーを更新することは、entity間の関連そのものを変更することなので、relationshipの削除と登録によって行なうものとした。

4. データ制御機能

(1)呼出し制御

プロジェクト構成員は、その人が所属するプロジェクトのリーダーによって、entity set、relationship set 毎に、図5に示すアクセス権の中の 하나가設定される。プロジェクト構成員の役割を、set 毎のアク

略号	意味
R	Read 可
RW	Read/Write 可
N	アクセスできない

〔図5〕 アクセス権

セス権のベクトルとして考え、これをその構成員のアクセス範囲と呼ぶ。このアクセス範囲は、ライフサイクルの進行につれて動的に変化し、各局面の作業内容に応じたdefault値をもつ。一例として、モジュール設計者のアクセス範囲を図6に示す。アクセス範囲により、組織全体が対象とするentity setとrelationship setのサブセットに対してのみ、アクセスを許すという制御を行なう。

また、個々のentityは、図7に示す状態をとり、その状態により、entityやrelationshipに対するアクセスが制限される。entityの状態遷移は、アプリケーション・プログラムが、データ制御機能であるSET BLUEとSET REDの処理をERDBMSに依頼することにより起こる。可能なentityの状態遷移を図8に示す。添字iは、そのentityに一段のrelationshipでつながっているentityの中で、状態が赤であるものの個数を示す。以下、SET BLUE、SET REDに伴う状態の遷移について説明する。

1. 状態が W_i であるentityをSET BLUEしたとき、 $W_i \rightarrow Y_i$ の遷移が起こる。

2. 状態が R_i であるentityをSET BLUEしたとき、 $R_i \rightarrow Y_i$ の遷移が起こる。この遷移に伴ない、一段関連するentityに対して、 $W_i \rightarrow W_{i-1}$ 、 $Y_i \rightarrow Y_{i-1}$ 、 $R_i \rightarrow R_{i-1}$ の遷移が起こる。

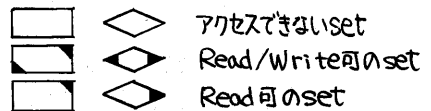
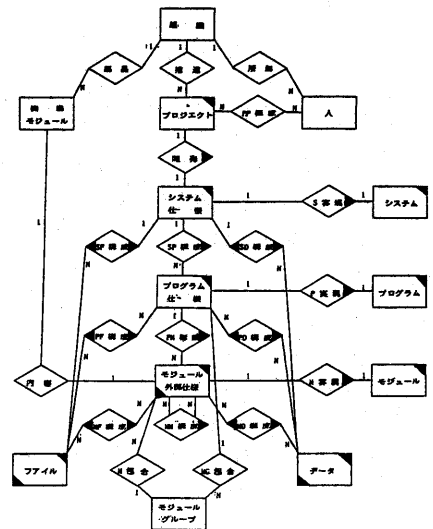
3. 状態が Y_i であるentityをSET REDしたとき、 $Y_i \rightarrow R_i$ の遷移が起こる。この遷移に伴ない、一段関連するentityに対して、 $W_i \rightarrow W_{i+1}$ 、 $Y_i \rightarrow Y_{i+1}$ 、 $R_i \rightarrow R_{i+1}$ の遷移が起こる。

4. 他の遷移は許されない。

上記3は、①変更entityに対するアクセスの制限を行ない、②関連entityに対するentityへのアクセスに警告を与え、変更による波及効果の制御を行なうためのものである。

呼出し制御は、以上述べてきた、①利用者のアクセス範囲、②アクセス対象の状態、と、③データ操作の種類、をそれぞれ軸とする3次元空間内で行なう。図9に呼出し制御の方針を示し、その基本的考え方を、以下に述べる。

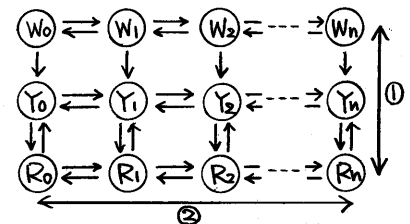
1. entityを参照する場合は、その対象とするsetに対して、Read可（Rと略す。）または、Read/Write可（RWと略す。）のアクセス権を必要とし、Rのときには、対象とするentityの状態が赤でないという制限が加わる。relation-



〔図6〕モジュール設計者のアクセス範囲

状態	意味
白	定義中である。
青	安定状態である。
黄	変更の可能性がある。
赤	変更中または変更必須である。

〔図7〕entityの状態



W_i : 白, Y_0 : 青, $Y_1 \sim Y_n$: 黄, R_i : 赤
①は、アプリケーション・プログラムが設定。
②は、ERDBMSが設定。

〔図8〕entityの状態遷移

shipの場合は、その対象とするset に対して、RまたはRWのアクセス権を必要とし、両側のentityに対して、前述した制限を受ける。この制限は、参照だけできる人は、アクセス対象が変更中のときは、参照できない」というルールを実現したものである。

2. entityを変更する場合、その対象とするset に対して、R、Wのアクセス権を必要とし、対象とするentityの状態が、白または、赤でなければならない。relationshipの場合は、その対象とするset に対して、RWのアクセス権を必要とし、両側のentity setに対して、Rまたは、RWのアクセス権を必要とし、entity key1 側のentityの状態が、白または、赤でなければならない。

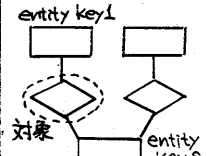
(2)同時実行制御

同時実行制御は、ERDBMS全体のロックと、個々のentityやrelationship単位のロックの2レベルで行なう。後者は、通常のアプリケーション・プログラムは、幾つかのデータ操作機能を組み合わせて、一連の処理を行なっているため、一つのデータ操作を実行する毎に、関連するentityやrelationshipを自動的にロックして、一連の処理を行なうのに支障がないように制御するためのものである。

データベース利用者の、対象とするset に対するアクセス権で、個々のentityや、relationshipをロックするモードが決まる。アクセス権がRWの人は、専有モードでロックし、他の人の使用を一切拒否し、Rの人は、共有モードでロックし、他の人の検索、読み出し等の変更を伴わないもののみを許す。図10に、可能なロックのモードの組み合わせを示す。○印はロック可能、×印はロック不可能を表す。

5. データ定義機能

ディクショナリ・エディタが提供するデータ定義機能を図11に示す。(1)、(2)は、内部スキーマを定義する機能で、(3)～(11)は、概念スキーマを定義する機能である。データベース管理者は、これらの定義言語を、端末より対話形式で入力し、データ定義を行なう。また、ディクショナリ・エディタは、これらの定義機能に加えて、すでに定義されているデータ定義情報を提示する機能、あるいは、それらをレポートとして出力する機能をもつ。

操作	対象set	アクセス権	色の状態	意味・コメント
検索 (RETRIEVE)	entity set	(1) R	赤でない	黄は禁止しないアプリケーション・プログラムが読める。
		(2) RW	*	修正担当者を想定している。
読み出し (GET)	relation- ship set	RまたはRW	両側のentity の状態が(1)ま たは(2)	
追加 (ADD)	entity set	RW	白または赤	白---開発中 赤---修生中(保守)
削除 (DELETE)	relation- ship set	RW 両側のentity setのアクセス 権は、Rまたは RW	entity key1 の状態は、白ま たは赤 entity key2 の状態は*	
更新 (UPDATE)				

〔図9〕 呼出し制御の方針 * don't care

ロックしようとするモード	他の人のロックの状態		
	ロックされてない	共有	専有
共有	○	○	×
専有	○	×	×

〔図10〕許されるロックのモードの組み合わせ

デ ー タ 定 義 語 の 構 文	
(1)	DEFINE ファイルタイプ, NAME=ファイル名, パラメータの並び
(2)	REDEFINE ファイル名, パラメータの並び
(3)	DEFINE セットタイプ, NAME=セット名, KEY=セット番号, パラメータの並び
(4)	REDEFINE セット名, パラメータの並び
(5)	DELETE SET セット名1, ---- セット名n
(6)	DEFINE ATTRIBUTE OF セット名, アトリビュート名1, アトリビュート番号1, valueセット番号1, アトリビュート長1, : アトリビュート名m, アトリビュート番号m, valueセット番号m, アトリビュート長m
(7)	DELETE ATTRIBUTE OF セット名, アトリビュート名1, ----, アトリビュート名m
(8)	DEFINE ENTITY OF relationshipセット名, entityセット名1, entityセット名2
(9)	DELETE ENTITY OF relationshipセット名, entityセット名1, entityセット名2
(10)	DEFINE DSET OF entityセット名, セット名1, ----, セット名n
(11)	DELETE DSET OF entityセット名, セット名1, ----, セット名n

〔図11〕 データ定義機能

以下、個々のデータ機能について、簡単に説明する。

(1)データベース編成に必要な種々のファイルを定義するもので、ファイルタイプには、定義するファイルの種類を記述する(図12)。ファイルの種類については、6章を参照。パラメータには、そのファイルのブロック長、レコード長、容量、等を記述する。

ファイルタイプ	ファイルの種類
ESF	entity file
RSF	relationship file
VSF	value file
EKF	entity key file
PRK	entity key1 file
SRK	entity key2 file

〔図12〕 ファイルタイプ

- (2)ファイル定義の変更を行なう。
 (3)種々のセットを定義するもので、セットタイプには、定義するセットの種類を記述する(図13)。
 (4)セット定義の変更を行なう。
 (5)セットを消去する。

セットタイプ	セットの種類
ENT	entity set
REL	relationship set
VAL	value set

〔図13〕 セットタイプ

(6)アトリビュートを定義するもので、value セット番号、アトリビュート長、等を記述する。

(7)アトリビュートを消去する。

(8)、(9)entityセットとrelationshipセットの関係を定義、消去する。

(10)、(11)あるentityセットに存在従属であるセットを定義、消去する。

6. データベース編成法

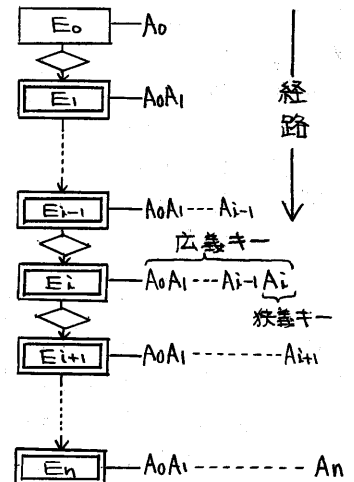
実体関連モデルに基づいた論理的なデータベース(概念スキーマ)を、物理的なデータベース(内部スキーマ)として、計算機上に構築する技法について述べる。これは、実体関連モデルによるデータ認識の4つのレベル⁽⁹⁾の中で、レベル2のentityとrelationshipに関する情報構造を、レベル3のアクセスパスに依存しないデータ構造へ、更に、レベル4のアクセスパスに依存するデータ構造へ変換する手法に対応する。ここでは、レベル3とレベル4とを区別しないで述べる。

内部スキーマの設計において、entity間の関係を、entityとは異なる列のタイプ (relationship set) として認識するというような、実体関連モデル自体のもつ性質と、そのモデル上に施すデータ操作の効率を考慮して、特に次の点に注意を払った。

- (1) 任意のentityを中心に考え、そのentityと一般のrelationshipで関係づけられているentityを、効率的に検索できること。
- (2) entityのキーを更新したとき、その影響が、関係するrelationshipのキーと、そのentityに存在従属であるすべてのentityのキーに及ばないこと。
- (3) 存在従属性を十分に表現できるデータ構造であり、存在従属性に起因するデータ処理を、効率的に行なえること。

以下、これらの条件を満たす技法について、詳細に説明するが、現在は、relationshipについては、二項関連のみを取り扱っているのので、これを中心に述べ、多項関連への拡張についても多少触れる。

説明上必要な用語を、幾つか準備する。entity set を E_i で表す。 E_i が E_{i-1} ($i=1, \dots, n$) に存在従属であるとき、 E_0 の主キーを A_0 とすれば、一般に、 E_i ($i=1, \dots, n$) の主キーは、 $A_0 A_1 \dots A_i$ で表される。ここで、 A_i は、 E_i のあるアトリビュートを示すものとする。entity set E_i に対して、 A_i を狭義キー、 $A_0 A_1 \dots A_i$ を広義キーと呼ぶ。また、 E_i に存するあるentityに対し、 $A_0, A_0 A_1, \dots, A_0 A_1 \dots A_{i-1}$ の値を、それぞれ広義キーとしてもつentityの集合を、もとのentityの経路 (path)と呼ぶ (図14参照)。 E_i が、他のどのentity setに対しても存在従属でないとき、広義キーと狭義キーは一致し、 E_i に属するentityは経路をもたない。



〔図14〕用語の説明

二項実体関連データベースの構成ファイルを図15に、構成ファイル間の関係を図16に示す。これらのファイルは、相対編成であり、セット毎にその容量に応じて分割され、すべてのレコードは、そのセット内における相対レコード番号をもつ。特に、entity file の場合に限り、これをオカーレンス番号と呼ぶ。

個々のentityやrelationshipは、1レコードの領域が割り当てられ、それぞれ entity file、relationship file に格納される。

entityの検索を速くし、また、狭義キーの更新を行なっても、そのentityのオカーレンス番号が変わらないように、狭義キーによる転置索引ファイル (entity

特徴 ファイル名	用途	相対編成	ちり編成 (割り算法 折りたたみ法 逐地開放)	転置索引 (部分転置)	つなぎ編成 (循環並び)
entity	entityの格納用	○			
entity key	entityの検索用	○	○	○	
relationship	relationshipの格納用	○			
entity key 1	entity間の関係	○	○	○	
entity key 2	の検索用	○	○	○	
value	アトリビュートの値の格納用	○			○

〔図15〕構成ファイルの特徴

key file)を別に用意した。entity key fileは、狭義キーによりハッシュされた、ただし編成ファイルである。ただし関数は、割り算法と折りたたみ法を併用したものを採用し、同義語(同一の番地にハッシュされる語)の処理には、番地開放方式を用いた。

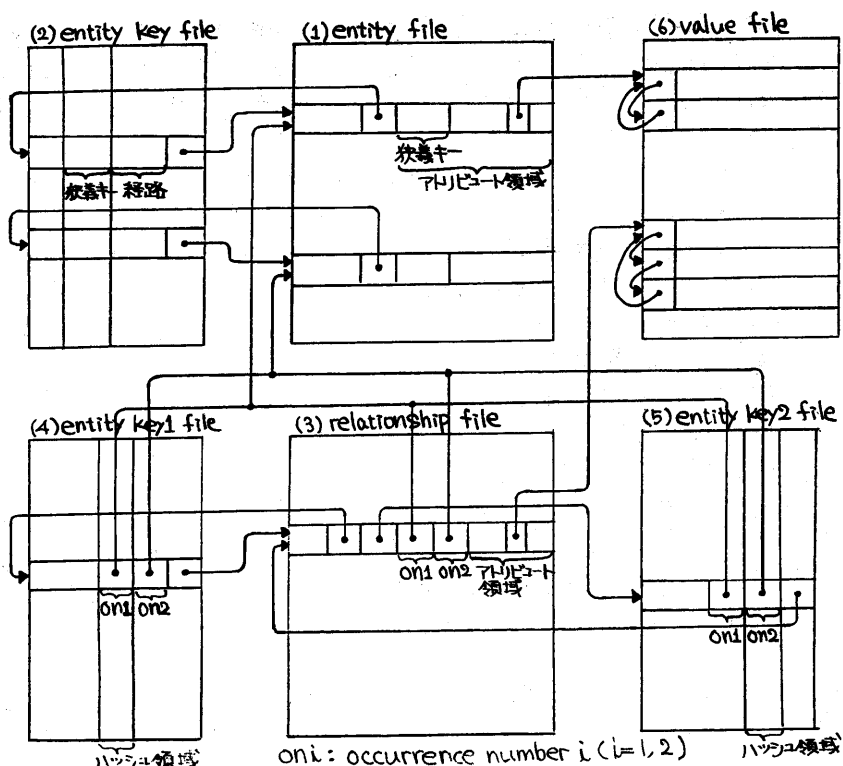
経路領域には、もしそのentityが他のentityに存在従属であるとき、そのentityの経路となるentityのオカーレンス番号を格納する。この

ため、あるentityの狭義キーが更新されても、その影響が、そのentityに存在従属であるすべてのentityに及ばなくなる。

entityの広義キーによる検索は、狭義キーによる転置索引ファイルを用いて検索開始点を見つけ、そこから順次、経路となるentityのオカーレンス番号に一致するものを捜せばよい。

relationshipの主キーとして、対応するentityの広義キーの値を格納せずに、そのentityのオカーレンス番号を格納する。このため、entityの狭義キーの更新の影響が、関係するrelationshipに及ばなくなる。

relationshipの検索を速くし、また、仕様のentityと一般のrelationshipで関係づけられているentityの検索を速くするために、relationshipの主キーを構成するentity key毎に、そのオカーレンス番号による転置索引ファイル(entity key1 file, entity key2 file)を用意した。ここで更に、主キーを構成する両方のentityのオカーレンス番号による転置索引ファイルを用意してもよいが、一般に n 項関連を取り扱う場合、 $2^n - 1$ 個の転置索引ファイルが必要になり、また、relationship登録時におけるオーバーヘッドも増え、実用的でない。そこで、主キーを構成する n 個のentityのオカーレンス番号による、 n 個の転置索引ファイルだけを用意し、relationshipの主キーによる検索は、どれか一つのentityのオカーレンス番号による転置索引ファイルを用いて、検索開始点を見つけ、そこから順次、残りのentityのオカーレンス番号に一致するものを捜すという方法を用いる。この方法は、多少検索の効率は落ちるが、実用的である。



〔図 16〕 構成ファイル間の関係

entityやrelationshipのアトリビュートの値は、検索の効率も考えて、検索条件に指定できるものは、直接そのentityやrelationshipに割り当てられたレコード内に格納し、指定できないものは、value set 内に格納する。検索条件として指定するアトリビュートの長さは、比較的短いものが多いので、この制限は妥当である。

以上述べてきた技法により、データベース編成における主キーの更新の問題、存在従属性の問題、等を、解決することができ、実体関連モデル上のデータ操作を効率的に行なうことが可能になった。

7. システムの評価

ERDBMSは、現在、FACOM 230-45S（主記憶 256KB）上で稼動しており、各アプリケーション・プログラムに対して、データベース・サービスを行なっている。ERDBMSのプログラムは、図17に示すように、コンパクトなものとなった。

	モジュール数	プログラム スライス数	主記憶サイズ
PL/I	68	4.8K	42KB
アセンブラ	11	1.1K	

〔図17〕 ERDBMSのプログラムの規模

データ操作機能の性能を図18に、各アプリケーションプログラムの性能を図19に示す。アプリケーション・プログラムのデータベースにアクセスするコマンドは、一回実行する毎に、データ操作機能を何回か実行するため、その応答時間は、データ操作機能の応答時間の数倍となっている。

性能は、OSのオーバーヘッド等、計算機システム的环境に深く依存する因子があるため、必ずしも十分とは言えないが、初期の要求は、達成できたと考えている。

データ操作機能		応答時間
OPEN	DBMS	2.3秒
CLOSE	DBMS	0.5
SET	PATH	2.6
RETRIEVE	OCCURRENCE	3.0
ADD	OCCURRENCE	3.1
DELETE	OCCURRENCE	3.8
GET	ATTRIBUTE	3.1
UPDATE	ATTRIBUTE	2.7

〔図18〕 データ操作機能の性能

アプリケーション ・プログラム名	応 答 時 間	
	データベースに アクセスしない コマンド	データベースに アクセスする コマンド
D S	1.0秒	12秒
S C G	1.0	6
R G	1.0	43
O G	1.0	10
P S	1.0	10

〔図19〕 各アプリケーションプログラムの性能

8. おわりに

以上、ERDBMSについて、その概要を述べてきたが、ERDBMSが提供する各機能は、プログラミング支援に十分に対応できる。今後、使用経験を積むことによりERDBMSを改良し、機能の強化を図り、性能向上に努めたい。現時点でERDBMSの機能強化として検討している点は、(1)非手続き的な高水準データ操作言語を提供して、アプリケーション・プログラムの労力を軽減させる。(2)多項関連をも扱うことのできるデータベース管理システムへ拡張する。(3)実体関連モデル上におけるview支援機能を提供する。(4)呼出し制御を一般化する。(5)データの正当性検証の機能を完備する。等で、汎用の実体関連モデルデータベース管理シ

システムの実現を目標としている。

本研究は、昭和55年度科学研究費補助金奨励研究(A)575226によって実施された。

[参考文献]

- (1) 落水：「プログラミング文書情報データベースによるプログラミング支援」。
情報処理学会 作譜工程の評価・改良・自動化シンポジウム報告集 昭和55年7月。
- (2) 落水：「ソフトウェア開発・保守支援システムPDB」
情報処理学会 ソフトウェア工学研究会 ソフトウェア工学17-5 昭和56年2月。
- (3) P. P. Chen：「The Entity-Relationship Model Toward a Unified View of Data」
ACM TRANS on Database Systems, Vol. 1, No. 1, March, 1978, pp9-36.
- (4) 「ADABAS入門」 ④ソフトウェア・エージー。
- (5) 植村俊亮：「データベースシステムの基礎」 オーム社。