(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第3763982号
(P3763982)
(45)発行日　平成18年4月5日(2006.4.5)　(24)登録日　平成18年1月27日(2006.1.27)

(51)Int.Cl.		F I		
G06F 12/00	(2006.01)	G06F	12/00	547Q
G06F 17/30	(2006.01)	G06F	12/00	513J
G06F 9/50	(2006.01)	G06F	17/30	170G
		G06F	17/30	180D
		G06F	17/30	180E

請求項の数 5　(全 18 頁)　最終頁に続く

(21)出願番号　特願平10-333585
(22)出願日　平成10年11月25日(1998.11.25)
(65)公開番号　特開2000-163307(P2000-163307A)
(43)公開日　平成12年6月16日(2000.6.16)
審査請求日　平成14年2月13日(2002.2.13)

(73)特許権者　000005108
株式会社日立製作所
東京都千代田区丸の内一丁目６番６号
(74)代理人　100083552
弁理士　秋田　収喜
(72)発明者　原　憲宏
神奈川県川崎市幸区鹿島田８９０番地　株式会社日立製作所　システム開発本部内
(72)発明者　土田　正士
神奈川県川崎市幸区鹿島田８９０番地　株式会社日立製作所　システム開発本部内

審査官　相崎　裕恒

最終頁に続く

(54)【発明の名称】データベース処理方法及びその実施装置並びにその処理プログラムを記録した媒体

(57)【特許請求の範囲】
【請求項１】
問い合わせ対象のデータ型に関連した関数を実現するモジュールを起動してデータベース処理を行うデータベース処理方法において、
ユーザからの問合せを解析する際に、ユーザからの問合せ中の関数を実現するモジュールのデータベースリソースアクセス情報が固有のデータベースリソースへのアクセスを行うことを示している場合には、当該データベースリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順を決定するステップを有することを特徴とするデータベース処理方法。
【請求項２】
当該モジュールのデータベースリソースアクセス情報が固有のデータベースリソースへのアクセスを行わないことを示している場合には、最もモジュール実行頻度の少ないデータベース処理装置にて当該モジュールを実行する様に前記実行手順を決定することを特徴とする請求項１に記載されたデータベース処理方法。
【請求項３】
当該モジュールのデータベースリソースアクセス情報が固有のデータベースリソースへのアクセスを行わないことを示している場合に、当該データベースリソースが格納管理されているデータベース処理装置とは関連の無いデータベース処理装置にて当該モジュールを実行する様に前記実行手順を決定することを特徴とする請求項１に記載されたデータベース処理方法。

【請求項４】
問い合わせ対象のデータ型に関連した関数を実現するモジュールを起動してデータベース処理を行うデータベース処理装置において、
データベースに定義されているデータ型に関連した関数を実現するモジュールが固有のデータベースリソースへのアクセスを行うかどうかを示すデータベースリソースアクセス情報を登録するモジュール定義情報登録部と、
ユーザからの問合せを解析する際に、ユーザからの問合せ中の関数を実現するモジュールのデータベースリソースアクセス情報が固有のデータベースリソースへのアクセスを行うことを示している場合には、当該データベースリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順を決定するモジュール実行手順決定処理部とを備えることを特徴とするデータベース処理装置。
【請求項５】
問い合わせ対象のデータ型に関連した関数を実現するモジュールを起動してデータベース処理を行うデータベース処理装置としてコンピュータを機能させる為のプログラムを記録した媒体において、
データベースに定義されているデータ型に関連した関数を実現するモジュールが固有のデータベースリソースへのアクセスを行うかどうかを示すデータベースリソースアクセス情報を登録するモジュール定義情報登録部と、
ユーザからの問合せを解析する際に、ユーザからの問合せ中の関数を実現するモジュールのデータベースリソースアクセス情報が固有のデータベースリソースへのアクセスを行うことを示している場合には、当該データベースリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順を決定するモジュール実行手順決定処理部としてコンピュータ機能させる為のプログラムを記録したことを特徴とする媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明はユーザがデータ型及びその振る舞いを定義することが可能なデータベース処理システムに関し、特にユーザ定義関数による固有のデータベースリソースへのアクセスを効率的に行うデータベース処理システムに適用して有効な技術に関するものである。
【０００２】
【従来の技術】
近年、データベース管理システムの分野では、ハードウェアの進歩やＷＷＷ(World Wide Web)の普及を背景に、文書や画像等のマルチメディアのデータをデータベースとして管理する需要が生じている。この為、従来のリレーショナル・データベース管理システムでは管理することができなかった複雑なデータを任意に定義し、管理する機能が必要になってきた。
【０００３】
その需要を満たすべく現在、データベース言語ＳＱＬ(Structured Query Language)ではＳＱＬ３(ISO,ANSI)の規格化が進められている。ＵＤＴ(User-Defined Type、ユーザ定義データ型)は、ＳＱＬ３の主要機能の１つである。ＵＤＴとは、ユーザ定義のデータ型であり、オブジェクト指向の概念を取り入れ、ＵＤＴデータに対する操作もメソッド（関数、手続き）としてユーザが定義する。ＵＤＴを用いると、複雑なデータ構造を実現することができる。また、ＵＤＴデータの値自身の振る舞い（付随する機能）をユーザ定義関数として定義することができる。ＵＤＴの定義では、値を表現する為の属性（群）の仕様と、その振る舞いを実現する一群の操作をＵＤＴ関数として規定する。これらＵＤＴ定義情報は、表定義等と同様にディクショナリ情報としてデータベース管理システムにおいて管理されるのが実装として一般的である。
【０００４】
上記オブジェクト指向の概念を取り入れ、ユーザ定義データ型等の機能をサポートしたリレーショナル・データベース管理システムを、オブジェクト・リレーショナル・データベ

ース管理システムと呼ぶことが近年多くなってきている。
【０００５】
ＵＤＴ関数の定義には、ＳＱＬ自身を用いた記述が可能である。また更に、この関数の定義には、Ｃ言語等の一般プログラミング言語で記述してコンパイルし、システムに登録したモジュールを指定することもできる。これらモジュールとＵＤＴ関数との関連は、ＵＤＴを定義する為のタイプ定義文（ＣＲＥＡＴＥ　ＴＹＰＥ文）に記述される。すなわち、モジュールとはＵＤＴ関数の内部的実現形態である。ＵＤＴ関数とモジュールの実装については、Don Chamberlin :"Using the new DB2:IBM's Object-Relational Database System", pp285,1996 等に記載されている。
【０００６】
ＵＤＴを使うことによって、マルチメディアデータに対応する機能をデータベース管理システムの機能として実現することができる様になる。これは、今までアプリケーションプログラムで行っていた処理を、データベースシステム側で高速にかつ低開発コストで実現できることを意味する。
【０００７】
【発明が解決しようとする課題】
ここでいうマルチメディアデータに対応する機能には、従来のリレーショナル・データベース管理システムには無い、複雑かつ高機能な検索機能が挙げられる。画像、音声、文書、地図情報等それぞれのデータに特有の検索システムには、それぞれの検索機能を満たす為のインデクスを有することが一般的である。オブジェクト・リレーショナル・データベース管理システムにおいても、ある種のインデクス若しくはそれに替わる情報を有し、ＵＤＴ関数を通じてそれらの「ＤＢリソース」にアクセスすることによりＵＤＴに特有の検索機能を提供することが必要となる。テーブルの列データであるＵＤＴデータに対する加工のみのＵＤＴ関数では、マルチメディアデータに対して十分ユーザニーズを満たす様な機能は実現できないからである。すなわち、マルチメディアデータに対するユーザのニーズに応えていく為にはテーブルデータの他に、ＵＤＴ下で管理されるＤＢリソースが必要となる。データベース管理システムにおいてその様なＤＢリソースを管理する様な仕掛けが必要となる。
【０００８】
一方、オブジェクト・リレーショナル・データベース管理システムの別の魅力は、リレーショナル・データベース管理システム上に存在する既存のデータをそのまま生かせること、すなわちリレーショナル・モデルのメリットである表形式検索にある。つまり、同一または別のテーブルにおける列データと関連した統合的な検索が可能であることである。
【０００９】
しかし、前述の様なＤＢリソースを有し、それに対するアクセスにより機能を実現する様なＵＤＴ関数を導入した場合、ＵＤＴ関数実現モジュールの実行（起動）場所がＤＢリソースの存在場所（サーバ）に限定される。よって、オブジェクト・リレーショナル・データベース管理システムのオプティマイザは、必ずＤＢリソースアクセスがあるかもしれない表格納ＢＥＳ（Ｂａｃｋ　Ｅｎｄ　Ｓｅｒｖｅｒ）にてＵＤＴ実現モジュールを起動する様にプランニングしなければならない為、複数行の突き合わせ処理を伴う問合せ要求等実行性能が十分に得られず、リレーショナル・モデルの効力を十分に享受できないという問題がある。
本発明の目的は上記問題を解決し、固有のＤＢリソースへアクセスするデータベース処理を効率的に行うことが可能な技術を提供することにある。
【００１０】
【課題を解決するための手段】
本発明は、問い合わせ対象のデータ型に関連した関数を実現するモジュールを起動してデータベース処理を行うデータベース処理システムにおいて、ユーザからの問合せ中の関数を実現するモジュールが固有のＤＢリソースへのアクセスを行う場合に、当該ＤＢリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順

を決定するものである。

【００１１】

本発明のデータベース処理システムでは、データベースに定義されているデータ型に関連した関数を実現するモジュールのモジュール定義情報を登録する際に、当該モジュールがそのデータ型に固有のＤＢリソースへのアクセスを行うかどうかを示すＤＢリソースアクセス情報を登録しておく。

【００１２】

ユーザからの問合せを受け取って対応するデータベース処理を行う際には、その問合せを解析してその問合せ中の関数を実現するモジュールのＤＢリソースアクセス情報を参照し、その情報が固有のＤＢリソースへのアクセスを行うことを示している場合には、当該ＤＢリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順を決定する。

【００１３】

また前記情報が固有のＤＢリソースへのアクセスを行わないことを示している場合には、最もモジュール実行頻度の少ないデータベース処理装置や当該ＤＢリソースが格納管理されているデータベース処理装置とは関連の無いデータベース処理装置にて当該モジュールを実行する様に実行手順を決定する。

【００１４】

以上の様に本発明のデータベース処理システムによれば、固有のＤＢリソースへアクセスするモジュールを当該ＤＢリソースが格納管理されているデータベース処理装置で実行し、それ以外のモジュールを他のデータベース処理装置で実行するので、固有のＤＢリソースへアクセスするデータベース処理を効率的に行うことが可能である。

【００１５】

【発明の実施の形態】

以下にユーザ定義関数による固有のデータベースリソースへのアクセスを効率的に行う一実施形態のデータベース処理システムについて説明する。

図１は本実施形態のデータベース処理システムの概略構成を示す図である。図１に示す様に本実施形態のデータベース管理システム１は、モジュール定義情報登録部６０と、ＵＤＴ関数実現モジュール実行手順決定処理部１１０と、実行手順指示作成処理部２１０とを有している。

【００１６】

モジュール定義情報登録部６０は、データベースに定義されているデータ型に関連した関数を実現するモジュールが固有のＤＢリソースへのアクセスを行うかどうかを示すＤＢリソースアクセス情報５２０を登録する処理部である。

【００１７】

ＵＤＴ関数実現モジュール実行手順決定処理部１１０は、ユーザからの問合せ中の関数を実現するモジュールのＤＢリソースアクセス情報５２０が固有のＤＢリソースへのアクセスを行うことを示している場合に、当該ＤＢリソースが格納管理されているデータベース処理装置で当該モジュールを実行する様に実行手順を決定する処理部である。実行手順指示作成処理部２１０は、ＵＤＴ関数実現モジュール実行手順決定処理部１１０での決定結果に応じて問合せ実行手順指示を作成する処理部である。

【００１８】

データベース管理システム１をモジュール定義情報登録部６０、ＵＤＴ関数実現モジュール実行手順決定処理部１１０及び実行手順指示作成処理部２１０として機能させる為のプログラムは、ＣＤ－ＲＯＭ等の記録媒体に記録され磁気ディスク等に格納された後、メモリにロードされて実行されるものとする。なお前記プログラムを記録する媒体はＣＤ－ＲＯＭ以外の他の媒体でも良い。

【００１９】

図１を用いて本実施形態の概念を簡単に説明する。本実施形態のデータベース管理システム１では、問合せ解析処理部１０において、ユーザ要求２のユーザ問合せに含まれるＵＤ

Ｔ関数実現モジュールに関するＤＢリソースアクセス情報５２０に応じて、ＵＤＴ関数実現モジュール実行手順決定処理部１１０でＵＤＴ関数実現モジュールをいつどこで実行するかを決定し、その決定結果を基に実行手順指示作成処理部２１０で問合せ実行手順指示を作成する。そして、データベース処理実行制御部２０において、問合せ実行手順指示に従ってモジュールを起動することによりＵＤＴ関数を実行する。

【００２０】

次に図１を用い、本実施形態のデータベース管理システムの構成について詳細に説明する。本実施形態のデータベース管理システム１は、ユーザからのデータベース問合せ要求であるＳＱＬ（構造化照会言語）を受け取り、構文解析、意味解析処理を通してデータベースアクセスの最適なアクセス経路を決定する最適化処理を行い、決定したアクセス経路に基づいてデータベース処理用の内部処理コード、すなわち問合せ実行手順指示を生成する問合せ解析処理部１０、生成された問合せ実行手順指示を基にデータベースアクセスの制御を行うデータベース処理実行制御部２０、データベース処理実行制御部２０の要求指示により、データベース領域アクセス処理部５０を通じてテーブルデータ４２へのアクセス制御を行うテーブルデータ管理部３０、また同様にデータベース処理実行制御部２０の要求指示により、ＵＤＴ関数実現ＵＤＴ関数実現モジュール９０の起動を行うモジュール起動部４０、テーブルデータ管理部３０若しくはモジュール起動部４０の要求指示により、ＤＢリソース４１及びテーブルデータ４２が格納されているデータベース領域４へのアクセス制御を行うデータベース領域アクセス処理部５０、ユーザから入力されるＵＤＴ関数実現モジュールに関するモジュール定義情報を受け付け解析し、ディクショナリ５への登録を要求するモジュール定義情報登録部６０、ユーザ要求が各種定義要求の場合、問合せ解析処理部１０の解析結果に基づきディクショナリ５に対する登録或いは削除を要求する定義処理部７０、ディクショナリ５に対する登録処理、参照処理或いは削除処理を行うディクショナリ管理部８０を有している。

【００２１】

ディクショナリ５には、テーブルやインデクスに関する定義情報、等の各種定義情報が格納されている。そのディクショナリ５に格納されている定義情報には、データタイプ定義情報５１、モジュール定義情報５２が含まれる。

【００２２】

データタイプ定義情報５１は、ＵＤＴに関する定義情報、すなわちＵＤＴを構成するデータ型及びそのＵＤＴの振る舞いを実現する為のＵＤＴ関数に関する情報である。データタイプ定義情報５１は、ユーザ要求２のデータタイプ定義要求によりユーザから入力されたものである。

【００２３】

そして、モジュール定義情報５２は、どのＵＤＴ関数の実現時にどのモジュールがどの契機で起動されるか等を示す情報である。更にモジュール定義情報５２には、そのモジュール起動時におけるＤＢリソースアクセスに関する情報であるＤＢリソースアクセス情報５２０を含む。モジュール定義情報５２は、モジュール定義情報３によりユーザから入力されたものである。

【００２４】

図２は本実施形態のコンピュータシステムのハードウェア構成の一例を示す図である。コンピュータシステム１０００は、ＣＰＵ１００２、主記憶装置１００１、磁気ディスク等の外部記憶装置１００３及び多数の端末１００４で構成される。主記憶装置１００１上には、図１を用いて先に説明したデータベース管理システム１が置かれ、外部記憶装置１００３上にはデータベース管理システム１が管理するデータベースに関する各種定義情報を含むディクショナリ５、及び定義されたテーブルデータ及びＵＤＴ関数実現モジュールがアクセスするＤＢリソースを含むデータベース領域４が格納される。また、データベース管理システム１を実現するプログラムも外部記憶装置１００３上に格納される。

【００２５】

図３は本実施形態のデータタイプの追加例を示す図である。ユーザがデータベース管理シ

ステムに対して新たなデータタイプを追加する際の手順について、図３の代表的な一手順例を用いて説明する。手順概略は以下の様になる。

【００２６】

（１）データタイプ定義（手順３０１）

（２）ＵＤＴ実現モジュール定義情報登録（手順３０２）

（３）テーブル定義（手順３０３）

まず、データタイプの定義（手順３０１）において、追加しようとするデータタイプに関する情報を登録する。データタイプの定義は、ＳＱＬの「ＣＲＥＡＴＥ　ＴＹＰＥ文」により行う。図３の「ＣＲＥＡＴＥ　ＴＹＰＥ文」の例では、text_no、text_name、author、contentsというそれぞれINT、CHAR、CHAR、VARCHAR型データの属性と、ＢＯＯＬＥＡＮ型を返すＵＤＴ関数ＣＯＮＴＡＩＮＳを有する「TEXT」型ＵＤＴが定義されている。

【００２７】

手順３０１のデータタイプ定義において定義したＵＤＴに関して、手順３０２においてＵＤＴ実現モジュール定義情報の登録を行う。以下に図３のデータタイプ定義に基づいたＵＤＴ実現モジュール定義情報の登録の際のユーザインタフェースの一例を挙げる。

【００２８】

```
ＵＤＴ名:TEXT
{
  _p_text_contains{
      起動契機          : AS_FUNCTION,
      ＵＤＴ関数名      : TEXT_CONTAINS
      ＤＢリソースアクセス : YES
  }
  _p_text_extract1{
      起動契機          : AS_FUNCTION,
      ＵＤＴ関数名      : TEXT_EXTRACT1
      ＤＢリソースアクセス : YES
  }
  _p_text_extract2{
      起動契機          : AS_FUNCTION,
      ＵＤＴ関数名      : TEXT_EXTRACT2
      ＤＢリソースアクセス : NO
  }
  _p_text_insert{
      起動契機          : AS_INSERT_TRIGGER
      ＵＤＴ関数名      : NULL
      ＤＢリソースアクセス : YES
  }
  _p_text_delete{
      起動契機          : AS_DELETE_TRIGGER
      ＵＤＴ関数名      : NULL
      ＤＢリソースアクセス : YES
  }
}
```

まず、関連するＵＤＴの名称が「ＵＤＴ名」にTEXTと示されている。その後に個々のモジュールに関する情報が指示される。モジュール"_p_text_contains"は、ＵＤＴ関数"TEXT_CONTAINS"を実現する為に起動されることを示している。ＵＤＴ関数実現の為のモジュールには起動契機として"AS_FUNCTION"を指定する。そして、「ＤＢリソースアクセス」YESは、ＵＤＴ関数実現の為のモジュールが起動実行時にそのＵＤＴ固有のＤＢリソースに対するアクセスを行うという宣言を示している。同様に、モジュール"_p_text_extract1"は、ＵＤＴ関数"TEXT_EXTRACT1"を実現する為に起動されることを示している。ＵＤＴ関数実現の為のモジュールには起動契機として"AS_FUNCTION"を指定する。本モジュールはＵＤＴ固有のＤＢリソースに対するアクセスを要することを示している。モジュール"_p_te

xt_extract2"は、ＵＤＴ関数"TEXT_EXTRACT2"を実現する為に起動されることを示している。ＵＤＴ関数実現の為のモジュールには起動契機として"AS_FUNCTION"を指定する。本モジュールはＵＤＴ固有のＤＢリソースに対するアクセスを行わないことを「ＤＢリソースアクセス」NOにより指定している。

【００２９】

モジュール"_p_text_insert"は、TEXT型データの挿入時（"AS_INSERT_TRIGGER"は挿入時が起動契機であることを示す）に起動されることを示している。また、モジュール"_p_text_delete"は、TEXT型データを含む行の削除時（"AS_DELETE_TRIGGER"は削除時が起動契機であることを示す）に起動されることを示している。モジュール"_p_text_insert"、"_p_text_delete"共にモジュール実行においてＤＢリソースに対するアクセスを要することを示している。

【００３０】

以上のデータタイプ定義（手順３０１）とＵＤＴ関数実現モジュール定義情報登録（手順３０２）によって、ＵＤＴ関数実現モジュールをデータベース管理システムが認知し、ユーザからの問合せ中のＵＤＴ関数に応じて対応するモジュールが適切な場所で起動されＤＢリソースをアクセスすることにより特有の機能を実現する為の情報が準備できたことになる。

【００３１】

上記ＵＤＴの定義の下に、以下の手順３０３でデータベース領域に格納するテーブルを定義する。テーブルの定義は、図３の手順３０３の例の様にＳＱＬにおいて「ＣＲＥＡＴＥ　ＴＡＢＬＥ文」により行う。この例では、title、country、produce_year、guide、movie_contentsというそれぞれCHAR、INT、DATE、TEXT、BLOB型の列から構成されるmovies_libテーブルが定義されている。また、title、country、publish_year、guide、book_contentsというそれぞれCHAR、INT、DATE、TEXT、BLOB型の列から構成されるbooks_libテーブルが定義されている。

【００３２】

図４は本実施形態のＵＤＴ関数実現モジュール定義情報５２の構成例を示す図である。図４ではディクショナリ５に格納されるＵＤＴ関数実現モジュール定義情報５２の構成の一形態を表している。ＵＤＴ関数実現モジュール定義情報は、既述の様にユーザ定義データタイプ（ＵＤＴ）を実現する為のモジュールに関する定義情報である。

【００３３】

図４に示す様に、ＵＤＴ毎にＵＤＴ関数実現モジュール９０の数のモジュール定義情報レコード５２１から成る。モジュール定義情報レコード５２１は、それぞれあるＵＤＴ関数実現モジュール９０に関連するモジュール名、ＵＤＴ名、起動契機、起動ＵＤＴ関数名、そしてＤＢリソースアクセスフラグにより構成される。

【００３４】

ＵＤＴ名５２３は、その関連するモジュール（モジュール名５２２により識別される）が、どのＵＤＴの振る舞いを実現する為に起動されるか、またはどのＵＤＴに対する操作を行った際に起動されるかを示している。

【００３５】

その関連するモジュールがＵＤＴの振る舞いを実現する為のもの、すなわちユーザ要求２の問合せに明示的に記述されるＵＤＴ関数を実現する為のものである場合、起動契機５２４にはユーザの指定に伴い"AS_FUNCTION"が設定され、起動ＵＤＴ関数名５２５にモジュール名５２２に関連するＵＤＴ関数名が示される。起動契機５２４が"AS_INSERT_TRIGGER"の場合、そのＵＤＴを含むテーブルデータが挿入された際に対応するモジュールが起動され、"AS_DELETE_TRIGGER"の場合、そのＵＤＴを含むテーブルデータが削除された際に対応するモジュールが起動される。また"AS_UPDATE_TRIGGER"の場合、テーブルデータに対する更新によってそのＵＤＴのデータが更新された際に対応するモジュールが起動される。

【００３６】

ここで、ＤＢリソースアクセスフラグ５２６は、モジュールが起動され実行する際に関連するＵＤＴ固有のＤＢリソースをアクセスするかどうかを示すフラグである。ＤＢリソースアクセスフラグがＯＮの場合、ＤＢリソースに対するアクセスを行うことを示す。またＯＦＦの場合、ＤＢリソースでのアクセスは行わないことを示す。

【００３７】

図５は本実施形態のモジュール定義情報登録部６０の処理手順を示すフローチャートである。図５では図３で説明したＵＤＴ関数実現モジュール定義情報登録（３０２）を実行するモジュール定義情報登録部６０における処理の流れの一例を表している。

【００３８】

ユーザから入力されたモジュール定義情報３より、１つのモジュール毎に次の処理を行う。まずステップ５０１において、登録対象であるモジュール定義情報の為の定義情報エントリを用意する。そして次の様にそのエントリに対し項目を設定していく。それらの項目の基になる情報は全てユーザから入力されたモジュール定義情報である。ステップ５０２においてモジュール名を設定し、ステップ５０３においてそのモジュールがどのＵＤＴの振る舞いを定義するものであるかを設定する。更にステップ５０４においてそのモジュールがどんな契機で起動されるかを入力情報に従い設定し、ステップ５０５でＵＤＴ関数名を設定する。

【００３９】

また更にステップ５０６において、ユーザから入力されたモジュール定義情報を基にＤＢリソースアクセス情報５２０であるＤＢリソースアクセスフラグ５２６を設定する。ステップ５０６においてユーザから入力されたモジュール定義情報の「ＤＢリソースアクセス」がYESかどうかを判定する。YESの場合にはステップ５０７に進み、ＤＢリソースアクセスフラグをＯＮに設定する。またNOの場合にはステップ５０８に進み、ＤＢリソースアクセスフラグをＯＦＦに設定する。双方の場合とも以上でエントリ項目設定は全て完了したので、ステップ５０９において設定したモジュール定義情報エントリをディクショナリ管理部８０のサービスを通してディクショナリへ登録する。もちろん、モジュール定義情報エントリに対して必要な項目が全て設定されれば良く、その設定順序はこの図５で示しているものと一致する必要はない。

【００４０】

前述のＵＤＴ実現モジュール定義情報の登録の際のユーザインタフェースの一例に従い、ディクショナリに登録したモジュール定義情報エントリの実際の例を図４で示している。モジュール"_p_text_extract2"は、ＤＢリソースアクセスがNOという定義なので、エントリの項目「ＤＢリソースアクセスフラグ」がＯＦＦになっている。

【００４１】

図６は本実施形態のＵＤＴ関数実現モジュール実行手順決定処理部１１０の処理手順を示すフローチャートである。図６では問合せ解析処理部１０でユーザ要求２（ＳＱＬ文）内にＵＤＴ関数が検出された場合のＵＤＴ関数実現モジュール実行手順決定処理部１１０の処理内容を表している。

【００４２】

ＵＤＴ関数の解析にて、関数ＤＢリソースアクセスフラグを基に、ＵＤＴ関数を実現するモジュールに最適な問合せ実行手順を生成する様に決定する。まず、ステップ１１２においてＳＱＬ文中のＵＤＴ関数の出現位置がどこかを判定する。ＵＤＴ関数の出現位置がWHERE句以降である場合に処理はステップ１１９に進み、以下の様なＵＤＴ関数に関する手順を決定する。

【００４３】

（ａ）ＵＤＴ関数起動時期：前

（ｂ）ＵＤＴ関数起動場所：各ＢＥＳ

すなわち、WHERE句の判定処理の「前」に判定対象データとしてテーブルデータが存在するＢＥＳ（Ｂａｃｋ　Ｅｎｄ　Ｓｅｒｖｅｒ）にてＵＤＴ関数実現モジュールを起動する様に決定する。各種スキャン（テーブルデータを逐次的に１つずつ取出すテーブルスキャ

ン、インデクスを用いて条件に合致するテーブルデータのみを取出すインデクススキャン等）により取出したテーブルデータに対し、ＵＤＴ関数実現モジュールを適応する。そして、その結果を判定する様にする。

【００４４】

ここでＢＥＳとはテーブルデータが格納されているデータベース領域を有し、テーブルデータへのアクセスを主に担当するデータベースサーバのことを言う。これに対し、主にユーザからの問合せを受付け、解析し、問合せ手順指示を作成し、ＢＥＳに問合せ実行を要求し、その結果をユーザに返すデータベースサーバのことをＦＥＳ（Ｆｒｏｎｔ　Ｅｎｄ　Ｓｅｒｖｅｒ）とここでは言う。また、ディクショナリを有し、各種定義情報の登録及び取得を担当するデータベースサーバをＤＳ（Ｄｉｃｔｉｏｎａｒｙ　Ｓｅｒｖｅｒ）と呼ぶことにする。複数のＢＥＳにテーブルデータを分割配置し、そのそれぞれのＢＥＳが並列にデータベースアクセスを行うことにより問合せに対する結果返却を高速に行う並列データベース管理システムにも上記サーバ構成は適応可能である。以上ステップ１２４にてＵＤＴ関数解析処理を完了する。

【００４５】

さて、ステップ１１２においてＵＤＴ関数の出現位置がSELECT句に続く場合、すなわち関数結果が射影対象になっている場合にステップ１１３に進み、統計情報により絞り込み結果量を推定する。ステップ１１４においてその絞り込み結果量が多いと推定される場合に処理はステップ１１５に進み、対応するモジュールのモジュール定義情報をディクショナリ管理部８０に要求して取得する。続いてステップ１１６においてモジュール定義情報内のＤＢリソースアクセスフラグの判定を行う。ＤＢリソースアクセスフラグがＯＮの場合に処理はステップ１２０に進み、以下の様なＵＤＴ関数に関する手順を決定する。

【００４６】

（ａ）ＵＤＴ関数起動時期：後

（ｂ）ＵＤＴ関数起動場所：各ＢＥＳ

すなわち、対応するＵＤＴ列のデータをテーブルデータから取出した「後」に、そのＵＤＴデータをＵＤＴ関数実現モジュールへのパラメータとしてそのＢＥＳにてＵＤＴ関数実現モジュールを起動する様に決定する。この様に手順を決定するのはＤＢリソースに対するアクセスが必要だからである。

【００４７】

複数表の突き合わせ処理の場合、まず各ＢＥＳにて絞り込みを行った後、ＪＳにてジョイン処理を行う。ＪＳは、ジョイン処理を担当するサーバであり、負荷分散の理由にてテーブルデータが存在するＢＥＳとは別のサーバに決定されることが多い。その後、ジョインの結果、最終的に絞り込まれたテーブルデータが存在する各ＢＥＳにて一旦場所を移してＵＤＴ関数実現モジュールを実行する。それはジョインの結果、最終的に絞り込まれるデータ量が十分少ないと推定されるからである。その為、各ＢＥＳへのＵＤＴデータの再転送は処理コストとして殆ど影響はないと考えられる。ＵＤＴ関数実現モジュールの結果を収集し、ＦＥＳに返却する様に決定する。以上ステップ１２４にてＵＤＴ関数解析処理を完了する。

【００４８】

さて、ステップ１１６においてモジュール定義情報内のＤＢリソースアクセスフラグがＯＦＦと判定された場合に処理はステップ１２１に進み、以下の様なＵＤＴ関数に関する手順を決定する。

【００４９】

（ａ）ＵＤＴ関数起動時期：後

（ｂ）ＵＤＴ関数起動場所：ＪＳ/ＦＥＳ

すなわち、対応するＵＤＴ列のデータをテーブルデータから取出した「後」に、最終絞り込み結果に対してＪＳ或いはＦＥＳにてＵＤＴ関数実現モジュールを起動する様に決定する。それは、ＤＢリソースに対するアクセスが不要であることからモジュール起動及びＵＤＴデータ転送を加味して柔軟にモジュール起動サーバを選定することができることを意

味する。
【００５０】
複数表の突き合わせ処理の場合、まず各ＢＥＳにて絞り込みを行った後、ＪＳにてジョイン処理を行う。その後、ジョインの結果、最終的に絞り込まれたテーブルデータに関してＵＤＴ関数実現モジュールをそのＪＳにて実行する。それはジョインの結果、最終的に絞り込まれるデータ量が十分少ない為、モジュール起動の総負荷が問題ないと推定されるからである。そしてその結果をＦＥＳに返却する様に決定する。以上ステップ１２４にてＵＤＴ関数解析処理を完了する。
【００５１】
さて、ステップ１１４においてＵＤＴ関数起動側の絞り込み結果量が少ないと推定される場合にステップ１１７に進む。ステップ１１７において、対応するモジュールのモジュール定義情報をディクショナリ管理部８０に要求して取得する。続いてステップ１１８においてモジュール定義情報内のＤＢリソースアクセスフラグの判定を行う。ＤＢリソースアクセスフラグがＯＮの場合に処理はステップ１２２に進み、以下の様なＵＤＴ関数に関する手順を決定する。
【００５２】
（ａ）ＵＤＴ関数起動時期：前
（ｂ）ＵＤＴ関数起動場所：各ＢＥＳ
すなわち、ＵＤＴ関数実現モジュールの起動は、最終的に絞り込みが行われる「前」に行う。そして対応するＵＤＴ列のデータをテーブルデータから取出した際に、そのＢＥＳにてそのＵＤＴデータをＵＤＴ関数実現モジュールへのパラメータとして起動する様に決定する。この様に手順を決定するのはＤＢリソースに対するアクセスが必要だからである。
【００５３】
複数表の突き合わせ処理の場合、まず各ＢＥＳにてＵＤＴデータを取出した後ＵＤＴ関数実現モジュールの起動を行う。その結果及び絞り込み結果をＪＳに転送し、最終的な絞り込み、すなわちジョイン処理を行う。そしてその結果をＦＥＳに返却する様に決定する。
【００５４】
データ取出し直後にＵＤＴ関数実現モジュールの起動を行う様に決定するのは、ＢＥＳにおけるテーブルデータの量が少なく、モジュール起動による総負荷が少ないと判断されるからである。また、ＵＤＴ関数実現モジュールは、ＢＥＳに存在するＤＢリソースをアクセスする必要があるからである。以上ステップ１２４にて、ＵＤＴ関数解析処理を完了する。
【００５５】
さて、ステップ１１８においてモジュール定義情報内のＤＢリソースアクセスフラグがＯＦＦと判定された場合に処理はステップ１２３に進み、以下の様なＵＤＴ関数に関する手順を決定する。
【００５６】
（ａ）ＵＤＴ関数起動時期：後
（ｂ）ＵＤＴ関数起動場所：ＪＳ/ＦＥＳ
すなわち、対応するＵＤＴ列のデータをテーブルデータから取出した「後」に、最終絞り込み結果に対してＪＳ或いはＦＥＳにてＵＤＴ関数実現モジュールを起動する様に決定する。それは、ＤＢリソースに対するアクセスが不要であることからモジュール起動及びＵＤＴデータ転送を加味して柔軟にモジュール起動サーバを選定することができることを意味する。
【００５７】
複数表の突き合わせ処理の場合、まず各ＢＥＳにて絞り込みを行った後、ＪＳにてジョイン処理を行う。その後、ジョインの結果、最終的に絞り込まれたテーブルデータに関してＵＤＴ関数実現モジュールをそのＪＳにて実行する。それはジョインの結果、最終的に絞り込まれるデータ量が十分少ない為、モジュール起動の総負荷が問題ないと推定されるからである。そしてその結果をＦＥＳに返却する様に決定する。以上ステップ１２４にてＵ

ＤＴ関数解析処理を完了する。

【００５８】

なおここでの統計情報による最適化の手法はあくまで一例であり、本実施形態において最適化の為の情報としてＤＢリソースアクセス情報を用いる点が重要である。

【００５９】

本ＵＤＴ関数の解析結果とその他のSQL文解析結果を基に、図１の実行手順指示作成処理部２１０にて実行手順を生成する。ここで、次の問合せ文を例にとって先に説明したUDT関数解析処理を具体的に説明する。

【００６０】

```
SELECT movies_lib.title,
       TEXT_EXTRACT1(movies_lib.guide,'本文'),
       TEXT_EXTRACT2(books_lib.guide)
  FROM movies_lib,books_lib
  WHERE TEXT_CONTAINS(movies_lib.guide,'independence') IS TRUE AND
        books_lib.publish_year=1997 AND
        movies_lib.title = books_lib.title
```

テーブルmovies_libのguide列とテーブルbooks_libのguide列はデータ型が、TEXT型のＵＤＴで定義されている。TEXT型データの値としては文書が格納されている。

【００６１】

関数TEXT_CONTAINSは、第一パラメータである指定TEXT型データ内に第二パラメータにより指定される文字列を含む場合にTRUE(BOOLEAN型)を返す関数である。関数TEXT_EXTRACT1は、第一パラメータである指定TEXT型データ内に第二パラメータにより指定されるタグ部分に相当する文書を抽出しVARCHAR型データとして返す関数である。関数TEXT_EXTRACT2は、第一パラメータである指定TEXT型データ内の文書データを全て返す関数である。その際タグ情報は省略される。図４のUDT関数実現モジュール定義情報からも分かる様に関数TEXT_CONTAINS,TEXT_EXTRACT1をそれぞれ実現するモジュール_p_text_contains及びモジュール_p_text_extract1は、実行時にTEXT型固有のＤＢリソースをアクセスする。この例では、ＤＢリソースには、タグ名称からそのタグに対応する要素番号を取得する為の索引が情報として格納されているものとする。

【００６２】

上記ＳＱＬ文の例は、movies_libテーブルのguide列（解説）に"independence"を含むテーブルデータのtitle（題名）と、books_libテーブルのpublish_year列（出版年）が1997であるテーブルデータのtitle（題名）とが一致するものに対して、そのmovies_libテーブルのtitle（題名）と、movies_libテーブルのguide列（解説）の中の「本文」と、books_libテーブルのguide列（解説）全文を返すという検索要求である。

【００６３】

上記ＳＱＬ文のＵＤＴ関数は図６で示したフローに従って次の様に解析される。まず、movies_libテーブルに対する部分の解析処理において、ＵＤＴ関数TEXT_CONTAINSは、WHERE句以降に出現することから図６のステップ１１９の様に決定される。またＵＤＴ関数TEXT_EXTRACT1に関しては、TEXT_CONTAINSによる絞り込みにより十分少なく絞り込まれると推定し図６のステップ１２２の様に決定される。すなわち、ＵＤＴ関数TEXT_EXTRACT1実現モジュールの起動は、ジョイン処理の前に各ＢＥＳにて行われる様に決定する。

【００６４】

次に、books_libテーブルに対する部分の解析処理において、ＵＤＴ関数TEXT_EXTRACT2に関しては、最終的にジョイン処理により絞り込み結果量が少なくなると推定し、更にＤＢリソースへのアクセスが不要であることから、図６のステップ１２３の様に決定される。

以上の解析結果を基に生成した実行手順指示の一例を図７及び図８に示す。

【００６５】

図７は本実施形態の実行手順指示の一例を示す図である。本ＳＱＬ文は２つのテーブルの突き合わせ処理を要するので、テーブルmovies_libに対するスキャン処理７２０、テーブルbooks_libに対するスキャン処理７３０、そして突き合わせ処理７１０の３つの部分から問合せ実行手順指示７００は構成される。図７の部分詳細７２０ａはテーブルmovies_libに対するスキャン処理７２０の詳細を表している。

【００６６】

図８は本実施形態のスキャン処理７３０及び突き合わせ処理７１０の詳細を示す図である。図８の部分詳細７３０ａはテーブルbooks_libに対するスキャン処理７３０の詳細を表しており、図８の部分詳細７１０ａは突き合わせ処理７１０の詳細を表している。

【００６７】

図９は本実施形態の問合せ実行処理の概要を示す図である。図９では図７及び図８に示した実行手順指示に従った問合せ実行処理の概略を表している。本概略図におけるデータベース環境は、テーブルデータ４２のmovies_libがＢＥＳ１０３ａとＢＥＳ１０３ｂに、テーブルデータ４２のbooks_libがＢＥＳ１０３ｄとＢＥＳ１０３ｅにそれぞれ分割格納管理されている並列データベース環境である。ＦＥＳ１０１、ＤＳ１０２、ＢＥＳ１０３は互いにネットワーク１０８により接続され、そのネットワーク１０８を介し問合せ要求、実行手順指示、中間加工結果及び問合せ結果を並列にやり取りする。

【００６８】

まず図７の問合せ実行手順指示の部分詳細７２０ａに従って、ＢＥＳ１０３ａとＢＥＳ１０３ｂそれぞれのサーバにおいて、テーブルmovies_libのテーブルデータを１件ずつアクセスし、guide列データ及びtitle列データを取出す。次にguide列データをパラメータにＵＤＴ関数実現モジュール９０の_p_text_containsを起動し、その結果を判定する。条件判定の結果、条件を満たしているなら、更に_p_text_extract1を起動する。本モジュールにおいて、ＤＢリソース４１へのアクセスが行われ、タグ「本文」に対応する部分を取得する。そして、図９のデータ１０４ａに示す様にその関数結果、titleデータ及びテーブルデータすなわち行を識別する行識別子をジョイン処理を行うＢＥＳ１０３ｃに転送する。以上の処理をテーブルデータがなくなるまで繰り返す。結果転送処理に関しては、転送に関するコストを考慮してある程度のデータ量をまとめて転送する方が望ましい。ジョインサーバであるＢＥＳ１０３ｃにおいて、ＢＥＳ１０３ａ及びＢＥＳ１０３ｂより転送されたデータ１０４ａをデータ１０４ｂとしてマージしておく。

【００６９】

上記テーブルmovies_libへのアクセス処理と同時に、図８の問合せ実行手順指示の部分詳細７３０ａに従って、ＢＥＳ１０３ｄとＢＥＳ１０３ｅそれぞれのサーバにおいて、テーブルbooks_libのテーブルデータを１件ずつアクセスし、guide列データ、publish_year列データ、及びtitle列データを取出す。次にpublish_year列に対する条件判定（=1997）を行い、条件を満たすならば図９のデータ１０５ａとして、titleデータ及びテーブルデータすなわち行を識別する行識別子をジョイン処理を行うＢＥＳ１０３ｃに転送する。以上の処理をテーブルデータがなくなるまで繰り返す。もちろん結果転送処理に関しては、転送に関するコストを考慮してある程度のデータ量をまとめて転送する方が望ましい。

【００７０】

ジョインサーバであるＢＥＳ１０３ｃにおいて、ＢＥＳ１０３ｄ及びＢＥＳ１０３ｅより転送されたデータ１０５ａをデータ１０５ｂとしてマージしておく。そして、ジョインサーバＢＥＳ１０３ｃにおいて、図８の問合せ実行手順指示の部分詳細７１０ａに従って、図９の突き合わせ処理１０６を行う。突き合わせ処理により更に絞り込まれたテーブルデータに関して、モジュール_p_text_extract2を起動する。最終的にその結果（TEXT_EXTRACT2結果）と転送されてきたTEXT_EXTRACT1の結果を問合せ結果１０７としてＦＥＳ１０１を経由してユーザに返す。

【００７１】

ジョインの結果絞られたデータ件数は非常に少ないので、ＤＢリソースアクセスを伴わないモジュール_p_text_extract2の起動をジョインサーバにおいて最終段階で行うことは、無駄な関数起動を行わない非常に効率の良い処理手順である。

【００７２】

以上示したフローチャートの処理は、図２で例として示したコンピュータシステムにおけるプログラムとして実行される。しかし、そのプログラムは図２の例の様にコンピュータシステムに物理的に直接接続される外部記憶装置に格納されるものと限定はしない。ハードディスク装置、フロッピーディスク装置等のコンピュータで読み書きできる記憶媒体に格納することができる。

【００７３】

前記の様に本実施形態のデータベース処理システムでは、ユーザ定義データ型の振る舞いを表すユーザ定義関数がそのＵＤＴ固有のＤＢリソースを効率的にアクセスすることを可能にしており、この為ユーザは多彩なＵＤＴを定義することができる。

【００７４】

以上説明した様に本実施形態のデータベース処理システムによれば、固有のＤＢリソースへアクセスするモジュールを当該ＤＢリソースが格納管理されているデータベース処理装置で実行し、それ以外のモジュールを他のデータベース処理装置で実行するので、固有のＤＢリソースへアクセスするデータベース処理を効率的に行うことが可能である。

【００７５】

【発明の効果】

本発明によれば固有のＤＢリソースへアクセスするモジュールを当該ＤＢリソースが格納管理されているデータベース処理装置で実行し、それ以外のモジュールを他のデータベース処理装置で実行するので、固有のＤＢリソースへアクセスするデータベース処理を効率的に行うことが可能である。

【図面の簡単な説明】

【図１】本実施形態のデータベース処理システムの概略構成を示す図である。

【図２】本実施形態のコンピュータシステムのハードウェア構成の一例を示す図である。

【図３】本実施形態のデータタイプの追加例を示す図である。

【図４】本実施形態のＵＤＴ関数実現モジュール定義情報５２の構成例を示す図である。

【図５】本実施形態のモジュール定義情報登録部６０の処理手順を示すフローチャートである。

【図６】本実施形態のＵＤＴ関数実現モジュール実行手順決定処理部１１０の処理手順を示すフローチャートである。

【図７】本実施形態の実行手順指示の一例を示す図である。

【図８】本実施形態のスキャン処理７３０及び突き合わせ処理７１０の詳細を示す図である。

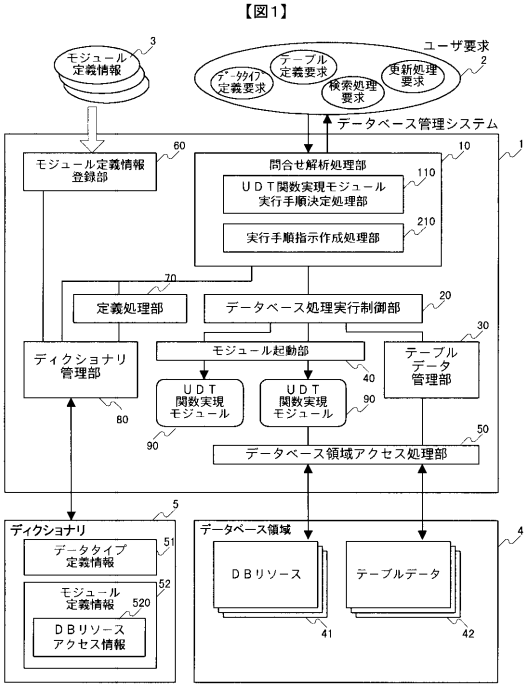
【図９】本実施形態の問合せ実行処理の概要を示す図である。

【符号の説明】

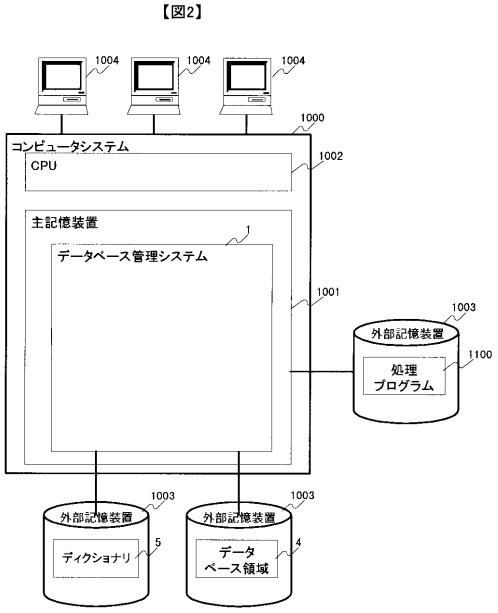
１…データベース管理システム、２…ユーザ要求、３…モジュール定義情報、１０…解析処理部、２０…データベース処理実行制御部、３０…テーブルデータ管理部、４０…モジュール起動部、５０…データベース領域アクセス処理部、７０…定義処理部、８０…ディクショナリ管理部、９０…ＵＤＴ関数実現モジュール、４…データベース領域、４１…ＤＢリソース、４２…テーブルデータ、５…ディクショナリ、５１…データタイプ定義情報、５２…モジュール定義情報、５３…統計情報、５２０…ＤＢリソースアクセス情報、６０…モジュール定義情報登録部、１１０…ＵＤＴ関数実現モジュール実行手順決定処理部、２１０…実行手順指示作成処理部、１０００…コンピュータシステム、１００１…主記憶装置、１００２…ＣＰＵ、１００３…外部記憶装置、１００４…端末、１１００…処理プログラム、３０１～３０３…手順、５２１…モジュール定義情報レコード、５２２…モジュール名、５２３…ＵＤＴ名、５２４…起動契機、５２５…起動ＵＤＴ関数名、５２６…ＤＢリソースアクセスフラグ、７００…問合せ実行手順指示、７１０…突き合わせ処理

、７２０及び７３０…スキャン処理、７２０ａ…部分詳細、７１０ａ及び７３０ａ…部分詳細、１０１…ＦＥＳ、１０２…ＤＳ、１０３…ＢＥＳ、１０４…データ、１０５…データ、１０６…突き合わせ処理、１０７…問合せ結果、１０８…ネットワーク。

【図１】

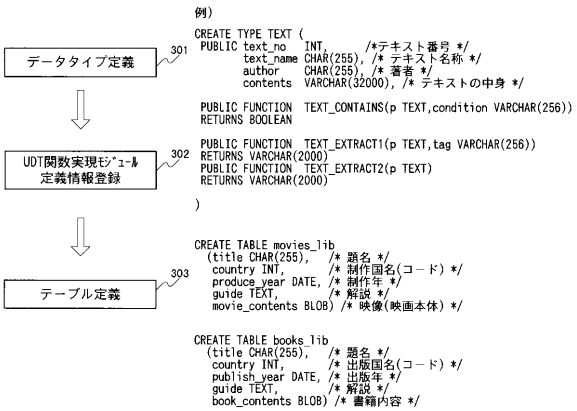


【図２】



【図３】

【図3】



【図４】

【図4】

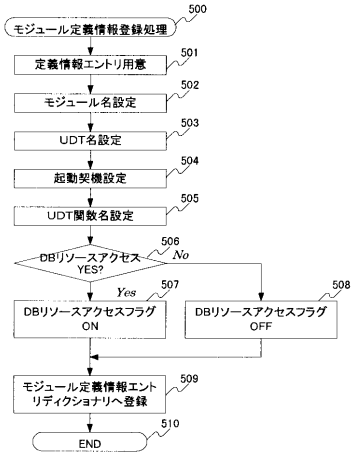
UDT関数実現モジュール定義情報 52

モジュール名 522	UDT名 523	起動契機 524	起動UDT関数名 525	DBリソースアクセスフラグ 521
p_text_contains	TEXT	AS_FUNCTION	TEXT_CONTAINS	ON
p_text_extract1	TEXT	AS_FUNCTION	TEXT_EXTRACT1	ON
p_text_extract2	TEXT	AS_FUNCTION	TEXT_EXTRACT2	OFF
p_text_insert	TEXT	AS_INSERT_TRIGGER	–	ON
p_text_delete	TEXT	AS_DELETE_TRIGGER	–	ON
…	…	…	…	…

526

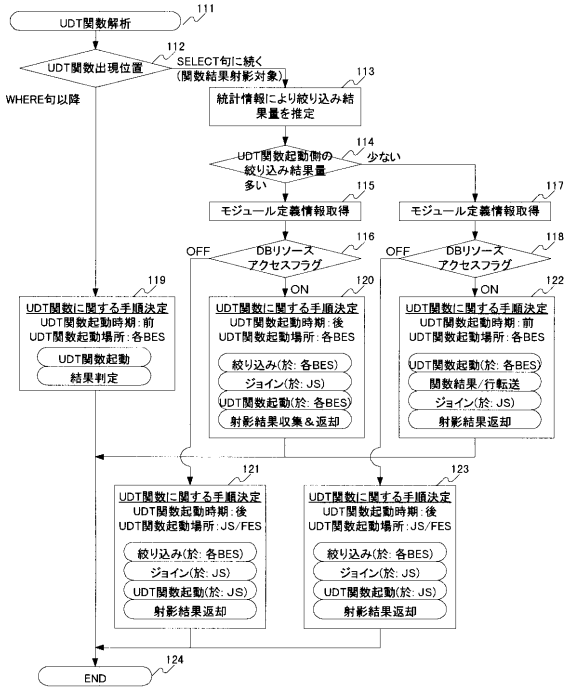
【図５】

【図5】



【図６】

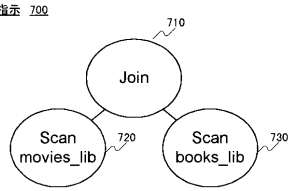
【図6】



【図７】

【図7】

問合せ実行手順指示 700



問合せ実行手順指示(部分詳細) 720a

Scan movies_lib

指示命令	情報	
繰返し実行	繰返しid	LOOP1
	取出しテーブルデータが無くなるまで繰返し	
スキャン (1テーブルデータアクセス)	テーブル	movies_lib
	取出しデータ	guide,title
関数起動	起動モジュール	_p_text_contains
	パラメータ情報	guide,'independence'
条件判定	IS. 関数TEXT_CONTAINS結果. TRUE	
関数起動	起動モジュール	_p_text_extract1
	パラメータ情報	guide,'本文'
結果転送	転送データ1	TEXT_EXTRACT1の結果 titleデータ 行識別子
	転送先	BES(103c)
繰返し終了	繰返しid	LOOP1

【図８】

【図8】

問合せ実行手順指示(部分詳細) 730a

Scan books_lib

指示命令	情報	
繰返し実行	繰返しid	LOOP2
	取出しテーブルデータが無くなるまで繰返し	
スキャン (1テーブルデータアクセス)	テーブル	books_lib
	取出しデータ	guide,publish_year,title
条件判定	=判定	publish_year 1997
結果転送	転送データ2	titleデータ 行識別子
	転送先	BES(103c)
繰返し終了	繰返しid	LOOP2

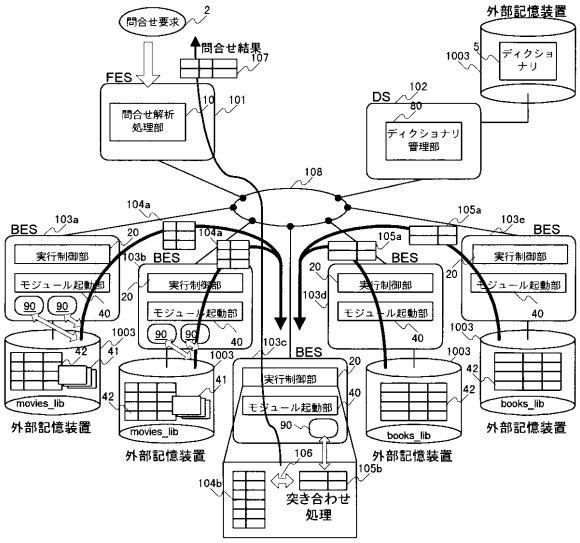
問合せ実行手順指示(部分詳細) 710a

Join

指示命令	情報	
繰返し実行	繰返しid	LOOP2
	突き合わせデータが無くなるまで繰返し	
付け合わせ処理	突き合わせデータ	movies_lib.titlebooks_lib.title
	突き合わせ条件	=条件
関数起動	起動モジュール	_p_text_extract2
	パラメータ情報	books_lib.guide
結果転送	転送データ3	movies_lib.title TEXT_EXTRACT1結果 TEXT_EXTRACT2結果
	転送先	FES
繰返し終了	繰返しid	LOOP2

【図９】

【図9】



フロントページの続き

(51)Int.Cl. FI
G06F 17/30 340Z
G06F 9/46 465C

(56)参考文献 特開平10－240588（JP，A）
Oracle8徹底解剖，日経オープンシステム，日本，日経BP社，1997年10月15日，p.216-225

(58)調査した分野(Int.Cl.，DB名)
G06F 12/00
G06F 17/30
G06F 9/50