

(19) 日本国特許庁 (JP)

(12) 特 許 公 報 (B2)

(11) 特許番号

特許第4483034号
(P4483034)

(45) 発行日 平成22年6月16日 (2010. 6. 16)

(24) 登録日 平成22年4月2日 (2010. 4. 2)

(51) Int. Cl.

F I

G O 6 F 12/00 (2006. 01)

G O 6 F 12/00 5 1 3 J

G O 6 F 17/30 (2006. 01)

G O 6 F 17/30 3 3 O B

G O 6 F 17/30 3 4 O Z

請求項の数 3 (全 18 頁)

(21) 出願番号 特願2000-174201 (P2000-174201)
 (22) 出願日 平成12年6月6日 (2000. 6. 6)
 (65) 公開番号 特開2001-350656 (P2001-350656A)
 (43) 公開日 平成13年12月21日 (2001. 12. 21)
 審査請求日 平成19年2月2日 (2007. 2. 2)

(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 100100310
 弁理士 井上 学
 (72) 発明者 猪原 茂和
 東京都国分寺市東恋ヶ窪一丁目280番地
 株式会社日立製作所中央研究
 所内
 (72) 発明者 西澤 格
 東京都国分寺市東恋ヶ窪一丁目280番地
 株式会社日立製作所中央研究
 所内

最終頁に続く

(54) 【発明の名称】 異種データソース統合アクセス方法

(57) 【特許請求の範囲】

【請求項 1】

第1のコンピュータと第2のコンピュータがネットワークで結合され、前記第2のコンピュータの持つ二次記憶には、それぞれが複数のカラムからなる複数のレコードで構成された第1のデータが保持され、該第1のデータに対してアプリケーションプログラムから発行される問合せを受け付ける第1のプログラムが前記第1のコンピュータに準備され、前記第1のデータの入出力を行う第2のプログラムが前記第2のコンピュータに準備されたコンピュータシステムにおけるデータアクセス方法であって、

前記第1のプログラムで受け付ける前記問合せは、前記第1のデータに含まれる1つもしくは複数のカラムに関する探索条件を含み、

前記第1のプログラムは、

前記問合せの受け付けに先立ち、前記第1のデータから、該第1のデータの前記複数のカラムの一部であり前記探索条件の対象となるカラムであるインデックスカラムと、該第1のデータの前記複数のカラムの一部であり前記第2のプログラムにアクセスするための引数となるカラムであるキーカラムとを組にした分散インデックスを抽出して保持し、

前記問合せを受け付けると、該問合せ中の前記探索条件を変形し、前記分散インデックスから前記探索条件に合致するレコード群のキーカラムを取得し、

該キーカラムを用いて前記第2のプログラム経由で前記第1のデータにアクセスすることにより、前記探索条件に合致するレコードを得て前記問合せの結果として前記アプリケーションプログラムに返送することを特徴とする請求項1記載のデータアクセス方法。

【請求項 2】

第 1 のコンピュータと第 2 のコンピュータがネットワークで結合され、前記第 2 のコンピュータの持つ二次記憶には、それぞれが複数のカラムからなる複数のレコードで構成された第 1 のデータが保持され、前記第 1 のデータに対してアプリケーションプログラムから発行される問合せを受け付ける第 1 のプログラムが前記第 1 のコンピュータに準備され、前記第 1 のデータの入出力を行う第 2 のプログラムが前記第 2 のコンピュータに準備されたコンピュータシステムにおけるデータアクセス方法であって、

前記第 1 のプログラムで受け付ける前記問合せは、前記第 1 のデータに含まれる 1 つもしくは複数のカラムに関する探索条件を含み、

前記第 1 のプログラムは、前記問合せの受付に先立ち、前記第 1 のデータから、該第 1 のデータの前記複数のカラムの一部であり前記探索条件の対象となるカラムであるインデックスカラムと、該第 1 のデータの前記複数のカラムの一部であり前記第 2 のプログラムにアクセスするための引数となるカラムであるキーカラムとの対応関係を示す分散インデックスを複数抽出して保持し、

前記アプリケーションプログラムは、前記問合せを、前記複数の分散インデックスのうちの該問合せで使用を許可する分散インデックスを指定する情報とともに発行し、

前記第 1 のプログラムは、

前記問合せを受け付けると、該問合せ中の前記探索条件を変形し、許可された分散インデックスから前記探索条件に合致するレコード群のキーカラムを取得し、

該キーカラムを用いて前記第 2 のプログラム経由で前記第 1 のデータにアクセスすることにより、前記第 1 の探索条件に合致するレコードを得て前記問い合わせの結果として前記アプリケーションプログラムに返答することを特徴とするデータアクセス方法。

【請求項 3】

第 1 のコンピュータと第 2 のコンピュータがネットワークで結合され、前記第 2 のコンピュータの持つ二次記憶には、それぞれが複数のカラムからなる複数のレコードで構成された第 1 のデータが保持され、前記第 1 のデータに対してアプリケーションプログラムから発行される問合せを受け付ける第 1 のプログラムが前記第 1 のコンピュータに準備され、前記第 1 のデータの入出力を行う第 2 のプログラムが前記第 2 のコンピュータに準備され、前記第 1 のプログラムは、前記問合せの受付に先立ち、前記第 1 のデータから該第 1 のデータの前記複数のカラムの一部であり前記問合せの探索条件の対象となるカラムであるインデックスカラムと、該第 1 のデータの前記複数のカラムの一部であり前記第 2 のプログラムにアクセスするための引数となるカラムであるキーカラムとを組にした分散インデックスを前記第 1 のコンピュータの二次記憶に格納するコンピュータシステムにおける前記分散インデックスの作成方法であって、

前記第 2 のコンピュータに準備された分散インデックス作成プログラムが、前記第 1 のプログラムから前記分散インデックスの作成要求を受け、

前記分散インデックス作成プログラムは、前記第 1 のデータから作成対象の分散インデックスのインデックスカラムとキーカラムを取り出し、取り出した結果を前記第 1 のプログラムに返答することを特徴とする分散インデックスの作成方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明はコンピュータシステムに関し、特に 1 つ以上のデータベースを用いてユーザの問合せを処理するデータ処理システムに関する。

【0002】

【従来の技術】

現在、企業の計算機システムには、多数のデータが存在している。これらのデータは、歴史的に企業の発展とともに目的別に追加されてきたものである。現在、業種間の規制緩和が急速に進展しており、これに伴い各企業は新規業務を追加していく傾向が強い。この際、新規業務の導入に伴って、さらに新たなデータが導入される場面が多くなっている。こ

10

20

30

40

50

これらのデータは、格納方法、形式などがまちまちである。例えば、リレーショナルデータベース管理システム中のデータベース、ファイルシステム中のフラットファイル、光磁気ディスクアーカイブ、表計算ソフトウェアのデータファイル等である。本明細書では、これらのデータ格納方法や形式のことをデータソースと呼ぶ。

【0003】

一方、規制緩和に伴い、各企業は他企業にない新たなサービスなどにより顧客によりよいサービスを提供し、その結果優良な顧客をより多く獲得しようと試みている。この際、多種のデータソース群に蓄積された過去の企業活動、顧客動向などを分析する必要性が高まり、データウェアハウスやデータマートの構築を行う企業が非常に多くなっている。

【0004】

データウェアハウスやデータマートの構築には、先に述べた多数のデータソースに蓄積されたデータを、ひとつの論理的に統合されたデータベースとすることが必要となる。また、データウェアハウスやデータマートのような分析処理の基盤となるデータベースを構築する以外にも、新規業務を迅速に立ち上げる目的で、従来のデータソース群を論理的に統合することが、企業の競争力を高める目的で必要とされている。論理的に統合したデータソース群を基盤とすることにより、新規業務のための応用プログラム（アプリケーション）構築の高速化を図ることが可能となるためである。

【0005】

データソースがデータベース管理システム（DBMS）の場合、情報基盤の統合をする方法として、データソース群とアプリケーション群の間に、DBMS群への統一的なアクセスを提供する「データベースハブ」のシステムを置く方法がある。データベースハブは、アプリケーションからの問合せ（典型的には、Structured Query Language（SQL）言語で記述された問合せ）を受けつけ、その問合せをDBMS群への問合せへ分解・変換する。そしてデータベースハブは、分解・変換した問合せをDBMS群に発行し、DBMS群から問合せ結果を作成するためのデータを収集し、アプリケーションの問合せに対する最終的な結果を得て、アプリケーションにその結果を返す。

【0006】

データベースハブを用いた情報基盤の統合は、以下の構成を取る。

【0007】

（１）ユーザアプリケーション（UAP）：データベースハブによって統合された情報を用いて処理を行うプログラム。

【0008】

（２）データベースハブ：１つ以上のデータソースを統合し、１つのデータベースとしてUAPに提供する。UAPからの問合せが複数のデータソースにまたがる場合、該複数のデータソースのデータを用いて、UAPからの問合せの結果を生成する。

【0009】

（３）データソース：統合対象となるデータを保持する。

【0010】

なお、データベースハブとデータソースは、多くの場合異なる計算機上に存在するが、同一の計算機上に存在しても差し支えない。

【0011】

【発明が解決しようとする課題】

データソースの一部分は、リレーショナルデータベース管理システム（RDBMS）であるが、その他のデータソースも用いられている。例えば、階層型データベース、ファイルシステム中のフラットファイル、光磁気ディスクアーカイブ中のファイル、表計算ソフトウェアのデータファイル等である。

【0012】

これらのデータソースの中には、RDBMSが登場する以前から存在していた基幹業務のデータであったり、データ量の問題でRDBMSに記録することが難しい（またはコストパフォーマンス上最適でない）データがある。しかし、これらのデータが、RDBMS中

10

20

30

40

50

に格納されているデータに比しても、戦略的重要度の高いデータである場合が少なくない。

【 0 0 1 3 】

これらのデータソースは、現在 R D B M S へのアクセスに広く用いられているデータベース問合せ言語 S Q L ではアクセスできないデータソースがほとんどである。また、上記のデータベースハブでは、データソースが R D B M S であること、即ちデータソース自身が S Q L を効率的に処理できることを前提として、S Q L を分解・変換する。

【 0 0 1 4 】

このため、データソースが S Q L を受けつけない場合、データベースハブからのアクセスでは、結果の指定をするために特定の探索条件（結果レコード群が満たすべき条件）を与える必要があるという制限があった。この特定の探索条件は、データソース中のデータを指定するためのキー情報である。このため、ユーザ（アプリケーション）からみると、自由な検索が困難で、アプリケーション開発時の負担が大きかった。また、この制限のため、定型業務には適用可能でも、非定型問合せが主体となる情報系業務への適用が困難だった。

10

【 0 0 1 5 】

また、データソースに S Q L でアクセスできてもアクセス効率が悪い場合、データベースハブを介した情報基盤の統合も、日常業務で現実的に使用することが困難なほど効率が悪くなる恐れがあった。これは、範囲検索等の多件数検索時に、データソースの全件検索に近いアクセスを余儀なくされ、ごく小規模以外の構成では現実的な性能を達成することが

20

【 0 0 1 6 】

本発明は、このような背景から、データソースが、R D B M S であっても、R D B M S 以外でも、情報基盤の統合を行うための技術を実現することを目的とする。

【 0 0 1 7 】

そこで、本発明が解決しようとする第 1 の課題は、非 R D B M S のデータソースを、R D B M S のデータソースと同じインタフェース（S Q L）でアクセスする際、非 R D B M S のデータソースを R D B データソースと同等の高い効率でアクセスすることにある。

【 0 0 1 8 】

第 1 の課題を解決するための手段として、後で述べる通り、非 R D B M S のデータソースから、該データソース中に格納されているデータの一部をインデックスとして取り出してデータベースハブに保持する。このインデックスを、従来の R D B M S 等で内部的に使用されているインデックスと区別する意味で、「分散インデックス」と称する。

30

【 0 0 1 9 】

非 R D B M S のデータソースとしては、戦略的重要度の高いデータを格納しているデータソースを、特に意識する。このようなデータソースの例としては、レガシーアプリケーションプログラム（レガシー A P）と、テープアーカイブや光磁気ディスクアーカイブ等の三次記憶が挙げられる。これらのデータソースでは、上記第 1 の課題の解決法である分散インデックスの作成に多大な処理時間がかかることが予想される。

【 0 0 2 0 】

そこで、本発明が解決しようとする第 2 の課題は、レガシー A P や三次記憶等、分散インデックス作成に多大な時間を要する恐れのある非 R D B M S データソースにおいても、分散インデックスを効率よく作成することにある。

40

【 0 0 2 1 】

また、分散インデックスは、データソースの一部を取り出してデータベースハブ側に保持するデータであるため、データソース側のデータが更新された場合、適切なタイミングでインデックスも更新する必要がある。

【 0 0 2 2 】

そこで、本発明が解決しようとする第 3 の課題は、データベースハブに対して、一旦作成したインデックスを管理するための方法を、データベースハブの管理者に提供することに

50

ある。

【 0 0 2 3 】

さらにデータソースによっては、データ量が莫大であるために R D B M S に保持することが困難なデータも含まれる。このようなデータソースに対しては、通常の R D B M S におけるインデックスのように全レコードに対する情報を保持することすら困難となる場合が想定される。例えば、光磁気ディスクアーカイブに格納されている数 T B (テラバイト) オーダーのデータは、インデックスとして必要なカラムを抽出したとしても数十 G B から数百 G B (ギガバイト) オーダーのデータになることも考えられる。一方で、このような大規模なデータの利用場面では、すべてのレコードを探索対象とするのではなく、特定の探索対象が設定されている場合が少なくない。そこで、本発明が解決しようとする第 4 の課題は、分散インデックスの対象レコードを利用場面に応じて絞り込み、分散インデックスが使用するデータ量を削減することである。

10

【 0 0 2 4 】

【課題を解決するための手段】

前記第 1 の課題を解決するため、本発明のシステムは、非 R D B M S のデータソースから、該データソース中に格納されているデータの一部をインデックスとして取り出してデータベースハブに保持する。このインデックスを、従来の R D B M S 等で内部的に使用されているインデックスと区別する意味で、「分散インデックス」と称する。分散インデックスは、データソースに対する探索条件を、データソースのレコード指定に対応づけるデータである。

20

【 0 0 2 5 】

データソースには、通常、1 つまたは複数のキーとなる情報が存在する。キーは、データソース中の、意味のあるひとかたまりのデータ (レコードと呼ぶ) を指定することができる情報である。多くの場合、キーによって、ただ 1 つのレコードを一意に指定することができる。また、多くの場合、キーによって指定したレコードに対して高速にアクセスする手段がデータソース側で提供されている。

【 0 0 2 6 】

例えば、顧客 I D がふられた顧客の情報を管理する顧客管理アプリケーションというデータソースがあったとする。この場合、顧客 I D をキーとして、顧客データ中のレコード (顧客 I D 、氏名、住所、年齢、電話番号、勤務先などの組) を特定することができる。

30

【 0 0 2 7 】

また、取引履歴データが、光磁気ディスクアーカイブに時系列で入っている場合を考える。ひとつひとつの取引情報が、時刻印とともに入っているとすると、時刻印をキーと考えることができる。この例では、時刻印によって完全に一意にひとつの取引情報を指定できるかどうかは、時刻印の与え方によるが、少なくとも時刻印をもちいることによって高速に 1 つの (またはたまたま同時刻に行われた少数の) 取引情報を得ることができる。

【 0 0 2 8 】

分散インデックスは、データソースに対する探索条件と、このようなデータソースのキーに対応づけるデータである。より具体的には、分散インデックスは探索条件の対象となっているデータ群と、キーとを組にして格納したデータである。探索条件を分散インデックスに対して適用することによって、探索条件を満たすキー群を得ることができる。このキー群を用いてデータソースにアクセスすることによって、データソースに対する高速なアクセスが実現できる。

40

【 0 0 2 9 】

従来の技術では、例えば、前記顧客管理アプリケーションが、「顧客 I D から顧客レコードを得る」というインタフェースのみを提供している場合、U A P からデータベースハブに「年齢が 3 0 才以上 4 0 才未満の顧客」という探索条件の問合せが発行されると、データベースハブが全顧客 I D を顧客管理アプリケーションに与えて全顧客レコードを得て、そこから該探索条件を全顧客レコードに適用して問合せの結果を得ていた。このため、データベースハブはデータソースである顧客管理アプリケーションから大量のレコードを入

50

手する必要がある、問合せの実行時効率が極めて悪かった。

【 0 0 3 0 】

本発明の分散インデックスを用いることにより、データベースハブは、まず分散インデックスに対して、「年齢が30才以上40才未満の顧客」という探索条件を適用して、この条件に合致する顧客ID群を得、これらの顧客IDを顧客管理アプリケーションに発行する、という方法で問合せの結果を得ることができる。この場合、「年齢が30才以上40才未満の顧客」に合致する顧客IDのみを顧客管理アプリケーションに対して発行すればよいので、顧客管理アプリケーションの処理量、およびデータベースハブと顧客管理アプリケーションとの通信が大幅に削減される。

【 0 0 3 1 】

分散インデックスを作成する際、データベースハブがデータソースの全レコードをアクセスすると、データベースハブとデータソースの間で大量の通信が発生する。この結果、分散インデックス作成時にネットワークおよびデータソースに多大な負荷がかかり、望ましくない。このため、本発明のシステムでは、データソースの存在する計算機に、インデックス作成プログラムを置く。インデックス作成プログラムが、該データソースの分散インデックスを一括して作成し、完成した分散インデックスをデータベースハブに転送する。これにより、分散インデックス作成時のデータベースハブとデータソースとの通信が1回で済み、ネットワーク負荷が大幅に軽減される。また、ネットワーク負荷の軽減にともない、データソースを保持する計算機のネットワーク処理負荷も大幅に軽減される。

【 0 0 3 2 】

分散インデックスは、RDBMS等が内部的に保持するインデックスと異なり、データソースに対する更新と連動して更新されない。このため、データベースハブのユーザおよび管理者が、分散インデックスを適切に利用、管理、運用するための手段が必要となる。このため、本発明のシステムでは、ユーザがどの分散インデックスを使用するか（もしくは使用しないか）を指定するインタフェースと、分散インデックスを作成し、最新のデータソースに合致させるインタフェースとを提供する。

【 0 0 3 3 】

既に述べた通り、データソースによっては、データ量が莫大であるためにRDBMSに保持することが困難なデータも含まれる。このようなデータソースに対しては、通常のRDBMSにおけるインデックスのように全レコードに対する情報を保持することすら困難となる場合が想定される。例えば、光磁気ディスクアーカイブに格納されている数TB（テラバイト）オーダーのデータは、インデックスとして必要なカラムを抽出したとしても数十GBから数百GB（ギガバイト）オーダーのデータになることも考えられる。このため本発明のシステムでは、分散インデックスとして、対象を全レコードではなく一部のレコードのみのキーを格納した分散インデックスを用いる。一部のレコードの選択方法としては、特定の探索条件を用いる方法、ランダムに選択によって選択を行う方法などを提供する。

【 0 0 3 4 】

これらの各手段によって、本発明のシステムはRDBMSのデータソースのみならず、レガシーAPや三次記憶等さまざまなデータソース中のデータを、1つのデータベースに格納されているかのようにユーザに提供し、かつ高い問合せ実行性能を実現することを可能にすることができる。

【 0 0 3 5 】

【発明の実施の形態】

本発明の実施の一形態を、図面を参照しながら説明する。

【 0 0 3 6 】

[1] 全体構成

図1を用いて、本発明の実施の一形態（実施例）の全体構成を説明する。

【 0 0 3 7 】

図1は、第1の実施例が好適に用いられるコンピュータシステムである。第1の実施例の

10

20

30

40

50

全体は、１つ以上のコンピュータ（データ処理システム１００、１つ以上のクライアントコンピュータ１０１、１０１'、...、管理用コンピュータ１０２、１つ以上のデータソース計算機１０５）が、クライアント側ネットワーク１０３およびサーバ側ネットワーク１０４で相互に接続されたコンピュータシステムである。

【００３８】

クライアント側ネットワーク１０３とサーバ側ネットワーク１０４はいずれも、ある団体（企業や学校や類似の団体）の全体や位置部門でよく使用されるＬＡＮでもよく、また地理的に分散した複数の地点を結合するＷＡＮの一部または全部でもよい。またこれらのネットワークは、計算機間結合網や並列計算機内部のプロセッサ要素間の結合網でもよい。また、クライアント側ネットワーク１０３とサーバ側ネットワーク１０４が同一のネットワークであっても差し支えない。

10

【００３９】

データ処理システム１００、クライアントコンピュータ１０１、１０１'、...、管理用コンピュータ１０２、データソース計算機１０５はいずれも、いわゆるパーソナル・コンピュータ、ワークステーション、並列計算機、大型計算機、小型携帯型コンピュータ等、任意のコンピュータでよい。

【００４０】

クライアントコンピュータ１０１、１０１'、...では、ユーザの処理を行うプログラムであるアプリケーション１２０、１２０'、...が動作する。アプリケーション１２０は、必要に応じてデータベースに対する参照または更新を、問合せを発行する。本実施例では、問合せ言語ＳＱＬで記述された問合せとする。

20

【００４１】

データソース計算機１０５は、データソース中のデータを保持し、他のプログラムのアクセスに応じてデータに対する参照または更新を行う計算機である。データソース中のデータに対する参照および更新の処理は、データソース入出力プログラム１２２が行う。データソース入出力プログラム１２２は、いわゆるレガシーＡＰでよい。データソース計算機１０５は多くの場合、その管理対象のデータを二次記憶装置１０６上に保持する。データソース計算機１０５、二次記憶装置１０６、データソース入出力プログラム１２２、およびその中に格納されているデータを総称して、データソース１０７と称する。なお、二次記憶装置１０６は、光磁気ディスクアーカイブ等、一般には三次記憶と称される記憶媒体でも差し支えない。

30

【００４２】

データソースのデータは、１つ以上の、意味のある塊をなしているものとする。この塊のひとつひとつを、ＲＤＢＭＳとの類似でレコードとよぶ。例えば、取引履歴というデータソースにおいて、１つの取引をレコードとみなすことができる。レコードがさらに複数のパーツからなる時、探索条件や出力項目として指定可能なパーツを、ＲＤＢＭＳとの類似でカラムと呼ぶ。例えば、１つの取引履歴レコードの中に「取引時刻」、「取引品名」などがある場合、これらをカラムとみなすことができる。例えば、データソース入出力プログラム１２２がいわゆるレガシーＡＰであっても、たとえば、「顧客ＩＤ」と、「住所」、「氏名」、「年齢」、「職業」とを関連づけて保持している場合、「顧客ＩＤ、住所、氏名、年齢、職業」を１つのレコード、「顧客ＩＤ」、「住所」、「氏名」、「年齢」、「職業」のそれぞれをカラムと考えて、なんら差し支えない。

40

【００４３】

データ処理システム１００は、クライアントコンピュータ１０１、１０１'、...の発行する第１の問合せを受け取り、必要に応じてデータソース１０７への１つ以上の第２の問合せを作成して発行し、第１の問合せが指定した参照または更新を行い、結果のデータを第１の問合せの発行元に返す。即ち、データ処理システム１００は、データソース１０７の保持するデータベース群への統一的なアクセスを実現し、クライアントコンピュータ１０１、１０１'、...へ統合されたデータベースを提供するデータベースハブである。

【００４４】

50

管理用コンピュータ 102 は、管理アプリケーション 121 を実行する。管理アプリケーション 121 は、データ処理システム 100 の管理を行うためのプログラムであり、典型的には、データ処理システム 100 または図 1 のシステム全体の管理者が利用する。

【0045】

入出力処理部 110、問合せ解析部 111、分散インデックス適用部 112、問合せ実行部 113、分散インデックス管理部 114、二次記憶装置 115 は、データ処理システム 100 を構成する構成要素である。これらの構成要素については、ここでは概略を説明するのに留め、動作の詳細については、あとで述べる。

【0046】

入出力処理部 110 は、クライアントコンピュータ 101, 101', ... からの問合せ要求、管理用コンピュータ 102 からの管理要求を受けつけるとともに、これらの要求に対する返答を行う。

10

【0047】

問合せ解析部 111 は、入出力処理部 110 が受けつけた問合せ要求の字句解析、構文解析、意味解析、を行い、必要に応じて問合せ条件の標準型変換を行い、問合せから構文解析木（パズツリー）を生成する。

【0048】

分散インデックス適用部 112 は、問合せ解析部 111 が作成したパズツリーを利用して、入力された問合せを、分散インデックスを用いるように変形する。この際、どの分散インデックスを利用するかを決定する必要があるが、この決定は分散インデックス管理部 114 が保持する個々の分散インデックスに関する管理情報を用いて行う。そして、問合せの結果を得るための一連の操作の手順（実行プラン）を生成する。リレーショナルデータベースの場合、一連の操作とは、選択処理、射影処理、ジョイン処理、グルーピング処理、ソート処理などである。実行プランは、これらの操作を、どのデータソース 107 のどのデータに対し、どの順番で適用するかを記述したデータ構造である。

20

【0049】

問合せ実行部 113 は、分散インデックス適用部 112 が生成した実行プランを実行する。問合せ実行部 113 はデータソース 107 への問合せを発行することにより、問合せを発行して前記一連の操作の一部または全部をデータソース 107 に依頼する場合もあるし、データソース 107 から取り寄せたデータに対し、自ら前記一連の操作の一部または全部を実行する場合もあってよい。

30

【0050】

分散インデックス管理部 114 は、入出力処理部 110 が受けつけた管理要求を解釈し、管理要求に含まれる分散インデックスの操作を行い、必要に応じて二次記憶装置 115 に保存する。また、分散インデックスに関する情報を保持し、分散インデックス適用部 112 がどの分散インデックスを適用するのが適当かを決定するのを支援する。

【0051】

以上が実施例の全体構成である。

【0052】

[2] データ構造

40

図 2 を用いて、分散インデックスの実現に用いるデータ構造について説明する。

【0053】

主に 2 種類のデータ構造を用いる。

【0054】

分散インデックス情報 210 は、データ処理システム 100 が保持する分散インデックスに関する情報を保持する。図 2 に示した分散インデックス情報 210 は、1 つの分散インデックスに対して保持する情報であり、データ処理システム 100 中に 1 つ以上存在する。

【0055】

インデックス ID 211 は、分散インデックスの名前である。インデックス ID 21

50

1 によって、各分散インデックスを一意に識別する。

【 0 0 5 6 】

対象データソース 2 1 2 は、該分散インデックスのもとになったデータソースである。後に述べるデータソース情報 2 2 0 のデータソース名 2 2 1 と対応する。

【 0 0 5 7 】

インデックスカラム 2 1 3 は、該分散インデックスが保持するカラム群である。分散インデックス適用部 1 1 2 は、このインデックスカラム 2 1 3 を用いて、ある探索条件を分散インデックスを用いて評価可能か否かを判定する。

【 0 0 5 8 】

キーカラム 2 1 4 は、該分散インデックスの対象データソースのキーである。ある探索条件を該分散インデックスを用いて評価した場合に、データソースへの問合せにおけるレコードの指定に用いるカラム群が何かを示す。キーカラム 2 1 4 のカラム集合は、インデックスカラム 2 1 3 のカラム集合に包含される。

【 0 0 5 9 】

インデックス格納テーブル 2 1 4 は、二次記憶装置 1 1 5 中に存在する該分散インデックスの実体の名前である。問合せ実行部 1 1 3 が分散インデックスを用いて探索条件の評価を行う場合には、インデックス格納テーブル 2 1 4 にアクセスする。

【 0 0 6 0 】

最終更新日付 2 1 5 は、該分散インデックスが最後に更新（データソースから作成）された時刻である。

【 0 0 6 1 】

データソース情報 2 2 0 は、データソース 1 0 7 に関する情報を保持する。図 2 に示したデータソース情報 2 2 0 は、1 つのデータソースに対して保持する情報であり、データ処理システム 1 0 0 中に 1 つ以上存在する。

【 0 0 6 2 】

データソース名 2 2 1 は、1 つのデータソースを一意に識別する名前である。

【 0 0 6 3 】

主キー 2 2 2 は、該データソースの主キーを保持する。主キーとは、該データソースにアクセス可能なカラム群を指す。データソースに対し、主キーを引数として指定したレコード参照（ここでは `getRecord`（主キー）と呼ぶ）が可能である。主キーは、物理的な格納順に対応したカラム群である場合が多い。主キー情報は、分散インデックスを自動的に作成する際のヒント情報として用いる。

【 0 0 6 4 】

分割 2 2 3 は、該データソースの分割方法（パーティショニング）の情報を保持する。大規模なデータソースの場合、物理的に複数の二次記憶装置に分割してデータを格納することにより、二次記憶装置の並列度を増したり、必要な容量を確保する。これがパーティショニングである。データソースの分割方法を活用する順序でアクセスを行うことにより、実行時間が大幅に改善されることが知られている。分割方法の情報も、分散インデックスを自動的に作成する際のヒント情報として用いる。

【 0 0 6 5 】

内蔵インデックス 2 2 4 は、該データソース内で、該データソースに定義しているインデックス群に関する情報を保持する。該データソース内部にインデックスがある場合、インデックスを利用した順序でアクセスを行うことにより、実行時間が大幅に改善されることが知られている。内蔵インデックスに関する情報も、分散インデックスを自動的に作成する際のヒント情報として用いる。

【 0 0 6 6 】

[3] 問合せに対する分散インデックスの適用

図 1 と図 3 とを用いて、分散インデックス適用部 1 1 2 が問合せに対して分散インデックスを適用する処理の流れを説明する。

【 0 0 6 7 】

アプリケーション 120 が発行した第 1 の問合せは、クライアント側ネットワーク 103 を経由してデータ処理システム 100 の入出力処理部 110 に到達する (150)。入出力処理部 110 は、入力アプリケーションからの問合せ要求であるか、管理用アプリケーションからの管理要求であるかを判定し、その結果に応じて、要求を問合せ解析部 111 へ送るか (151) 分散インデックス管理部 114 へ送る (160)。

【0068】

問合せ解析部 111 が第 1 の問合せを受け取ると、第 1 の問合せの字句解析、構文解析、意味解析を行う。この一連の処理により、第 1 の問合せから第 1 のパースツリーを生成する。なお、字句解析、構文解析、意味解析の動作については、コンパイラ、データベース管理システムなど多くの分野で用いられている技術であるため、ここではこれ以上詳細には述べない。

10

【0069】

問合せ解析部 111 は、第 1 のパースツリーを分散インデックス適用部 112 へ送る (152)。

【0070】

分散インデックス適用部 112 では、第 1 のパースツリーを検査し、分散インデックスが適用可能かどうかを判定する。図 3 の処理である。

【0071】

図 3 で示す一連の処理で問合せの探索条件を処理する。探索条件とは、データソースの一群のレコードを絞りこむための指定である。SQL 言語では、WHERE 句や HAVING 句などがこれにあたる。

20

【0072】

ステップ 301 で、探索条件を CNF 変換する。CNF (Conjunctive Normal Form) とは、探索条件の要素がまず OR で接続され、それらの接続が AND で接続された形式である。例えば、「 $(c1=10 \text{ and } c2=20) \text{ or } c3=30$ 」の CNF 変換の結果は、「 $(c1=10 \text{ or } c3=30) \text{ and } (c2=20 \text{ or } c3=30)$ 」となる。すべての結果レコードが、CNF 変換後の探索条件の各 OR 接続条件を満たすという性質がある (上記の例では、「 $c1=10 \text{ or } c3=30$ 」と「 $c2=20 \text{ or } c3=30$ 」が OR 接続条件)。

30

【0073】

ステップ 302 で、探索条件について、データ処理システム 100 が保持する各分散インデックスを検査する。すべての分散インデックスを検査したら (判定 Y)、分散インデックス適用の処理を終了する。

【0074】

ステップ 303 で、分散インデックスを 1 つ取り出す。ここで、該分散インデックスを X と呼ぶ。

【0075】

ステップ 304 で、X に対応する分散インデックス情報 210 の対象データソース 212 を参照 ((153)) して X の対象データソースを得て、探索条件を検査することにより、X の対象データソースが探索条件に含まれるか否かを判定する。含まれれば (判定 Y) ステップ 305 に制御を移し、含まれなければ (判定 N)、ステップ 302 に制御を移す。

40

【0076】

ステップ 305 で、探索条件中に含まれる X の対象データソースから、対象データソースを 1 つ選択する。選択したデータソースを Y と呼ぶ。このステップでは、1 つの問合せ中で 1 つのデータソースが複数回参照される可能性を考慮している。例えば、「SELECT \times FROM T1 A, T1 B WHERE $A.C1=B.C2$ 」という問合せでは、T1 というデータソースが 2 回、A と B という名前で登場している。

【0077】

50

ステップ306で、探索条件中の各OR接続条件に着目した場合に、該OR接続条件中で使用するデータソースYのカラム集合が分散インデックスXのカラム集合によって含まれているか否かを検査する。包含している場合(判定Y)、ステップ307に制御を移し、包含していなければ(判定N)、ステップ305に制御を移す。分散インデックスXのカラム集合は、Xのインデックスカラム213に格納されている。

【0078】

ステップ307では、分散インデックスXのカラム集合によって包含されているOR接続条件を、Xを用いた探索条件に書換える。具体的には、もともとT1にかかっていた探索条件を分散インデックスXに対して適用してキー(X.key)を得、該キー集合を用いてT1にアクセスし、結果レコードを得る、という問合せに書換える。例えば、Xのインデックスカラム213がT1.C1を含む場合、「SELECT x FROM T1, T2 WHERE T1.C1=10」を、「SELECT x FROM T1, T2 WHERE T1.key in (SELECT X.key FROM X WHERE X.C1=10)」とする。

【0079】

ステップ308では、すべてのYを検査したか否かによって、ステップ305またはステップ302に制御を移し、繰り返しを続ける。

【0080】

以上の一連の処理により、入力された問合せを、分散インデックスを利用した問合せに書換えることができる。

【0081】

図1に戻り、分散インデックス適用部112の残りの部分の処理を説明する。分散インデックス適用部112ではさらに、問合せ解析部111から得た第1のパーズツリーを用いて、問合せ最適化を行い、第1の問合せの実行プランを作成する。なお、場合によっては、第1の問合せ動作指示以外に追加の問合せ動作指示を得る必要がある場合がある。例えば、コストベース最適化の中間段階で表のレコード数が判明し、このレコード数を持ちいて問合せ分類定義を検索し、新たな問合せ動作指定を得る場合である。この場合の問合せ動作指定の取得方法は、前記問合せ照合処理と同様であるため、特に改めて説明はしない。

【0082】

第1の問合せの実行プランは、コストベース最適化により作成するが、コストベース最適化は文献1等ですでに広く知られているため、コストベース最適化の詳細についてはここでは述べない。

【0083】

分散インデックス適用部112が生成した実行プラン(第1の実行プラン)の例をひとつ挙げる。以下のリスト表現で表されるツリーである：(database hub join [left.c1=right.c2 and left.c3<10, output left.c1, right.c2, left.c1+left.c3] (join at DBMS1 [left.c1<10 and left.c1=right.c4, output left.c1, left.c3] (selection at DBMS1 CustomerTable [1990<year and year<1999, output c1, c3]) (selection at DBMS1 ProductTable [1000<price and price<2000, output c4])) (selection at DBMS2 OrderTable [1990<year and year<1999, output c2])) この実行プランは、『(1)DBMS1でCustomerTableに対し、探索条件「1990<year and year<1999」の選択処理を行い、射影処理によってカラムc1とc3を出力し、(2)DBMS1でProductTableに対し、探索条件「1000<price and price<2000」の選択処理を行い、射影処理によってカラムc4を出力し、(3)DBMS2でOrderTableに対し探索条件「1990<

10

20

30

40

50

year and year<1999」の選択処理を行い、射影処理によってカラム c2 を出力し、(4) DBMS1 でジョイン条件「left.c1<10 and left.c1=right.c4」((1) の中間結果が left、(2) の中間結果が right とする) でジョインを行って、射影処理によってカラム c1、c3 を出力し、(5) データ処理システム 100 でジョイン条件「left.c1=right.c2 and left.c3<10」((4) の中間結果が left、(5) の中間結果が right とする) のジョインを行い、射影処理により left.c1, right.c2, left.c1+left.c3 を出力する』という一連の処理を表現している。

【0084】

分散インデックス適用部 112 は、生成した第 1 の実行プランを問合せ実行部 113 に送る(154)。

10

【0085】

問合せ実行部 113 は、分散インデックス適用部 112 から得た第 1 の実行プランを用いて、第 1 の問合せの実行を行う。問合せ実行部 113 は、上述の例の第 1 の実行プランを、ボトムアップに、即ち上記(1)、(2)、(3)、(4)、(5)の順に処理していく(正確には、(1)、(2)、(3)は並列に実行することが可能である)。問合せ実行部 113 が最終的に実行プランに定められたすべてのステップを実行し、第 1 の問合せに対する最終的な結果が得られると、該結果は第 1 の問合せを発行したアプリケーション 120 へ入出力処理部 110 を経て返される(155、155'、156、156' および 157)。

20

【0086】

以上が、分散インデックスの適用を含む問合せ処理の流れである。

【0087】

[4] 分散インデックス利用を含む問合せの実行

分散インデックスを利用する問合せは、基本的には上記の問合せ実行部 113 の処理で述べた通りであるが、1つの分散インデックスが探索条件中に複数回登場する場合には、より効率的な実行方法を取ることができる。この手順を図4を用いて説明する。

【0088】

ステップ401で、1つの分散インデックスを用いた複数のOR接続条件(c1, c2, ..., cNとする)を得る。これらc1, c2, ..., cNを実行し、それぞれ結果を得る。この結果を、K1, K2, ..., Knとする。K1, K2, ..., Knはそれぞれ、該分散インデックスの対象データソースのキーの集まりである。

30

【0089】

ステップ402で、K1, K2, ..., Knの共通部分Kを得る。ただし、この共通部分は、SQLにおける"INTERSECT ALL"である。

【0090】

ステップ403で、Kに含まれるキーのそれぞれについて、該分散インデックスの対象データソースに対し、getRecord(key)を発行する。ここで、getRecord(key)は、対象データソース中でキー値がkeyのレコードを参照する、データソース107への呼び出しである。この一連の呼び出しで得たレコード群を結果表とする。

40

【0091】

ステップ404で、結果表に対して、まだ処理していない探索条件を実行する。

【0092】

この一連の処理により、複数のOR接続条件にまたがった絞り込みを一括して分散インデックスで処理し、しかるのちにデータソースにアクセスする、というアクセス方法が実現できる。このアクセスは、各OR接続条件を個々に処理する方法に比べ、データソースへのアクセス回数を大幅に削減できる可能性がある。

【0093】

50

〔 5 〕分散インデックスの作成

図 5 と図 6 を用いて、分散インデックス作成の処理の手順を説明する。

【 0 0 9 4 】

ここで説明する処理は、分散インデックス作成の 3 種のインタフェースである。これらのインタフェースは、管理用アプリケーションが用いるインタフェースであり、入出力処理部 1 1 0 が管理用アプリケーションからの要求を受付け、要求を分散インデックス管理部 1 1 4 へ送った場合 (1 6 0) に起動される。なお、本実施例ではアプリケーション 1 2 0 と管理アプリケーション 1 2 1 を区別しているが、これらを、双方の機能をあわせ持ったアプリケーションプログラムとして実現しても差し支えない。

【 0 0 9 5 】

分散インデックス作成の第 1 のインタフェースは、`createDistributedIndex` (対象データソース、キーカラム、インデックスカラム) という形式である。第 2 のインタフェースは、キーカラムを省略した、`createDistributedIndex` (対象データソース・インデックスカラム) という形式である。第 3 の形式は、キーカラム、インデックスカラムともに省略した `createDistributedIndex` (対象データソース、インデックスタイプ) という形式である。インデックスタイプは、「主キー優先」、「分割優先」、「内蔵インデックス優先 (内蔵インデックス名) 」の 3 種がある。これら 3 種のインタフェースは、完全に管理者が指定した分散インデックスを生成する方法から、データ処理システム 1 0 0 が半自動で分散インデックスを生成する方法までをカバーする。

【 0 0 9 6 】

ステップ 5 0 1 からステップ 5 0 6 で、3 種のインタフェースをサポートする。まずステップ 5 0 1 で、キーカラムが指定されたか否かによって、ステップ 5 0 2 またはステップ 5 0 3 に分岐する。

【 0 0 9 7 】

ステップ 5 0 2 では、第 1 のインタフェースに従って、指定されたキーカラムを用いて分散インデックスの作成を進める。

【 0 0 9 8 】

ステップ 5 0 3 では、すでに参照可能なデータソース情報 2 2 0 がデータ処理システム 1 0 0 中に存在しているか否かによって、ステップ 5 0 4 またはステップ 5 0 5 に分岐する。データソース情報 2 2 0 が存在している場合、5 0 4 でデータソース情報 2 2 0 の主キー 2 2 2 を新規に生成する分散インデックスのキーカラムとする。

【 0 0 9 9 】

また、データソース情報 2 2 0 が存在していない場合、分散インデックス管理部 1 1 4 が該データソースに対しアクセスを行い、キーカラムの情報 (および分割およびインデックスが存在していればこれらの情報) を取得する。取得できない場合はエラーとなる。そして、主キーをキーカラムに設定する。

【 0 1 0 0 】

5 0 6 では、インデックスカラムが決定していない場合、インデックスカラムを決定する。インデックスカラムの決定を要するのは、第 3 のインタフェースであるので、「主キー優先」、「分割優先」、「内蔵インデックス優先 (内蔵インデックス名) 」のいずれかによって、データソース情報 2 2 0 の主キー 2 2 2、分割 2 2 3、内蔵インデックス 2 2 4 のいずれかを参照し、分散インデックスのインデックスカラムを決定する。決定したキーカラム、インデックスカラムを、分散インデックス作成対象のデータソースに存在する分散インデックス作成部 1 2 3 に送る (1 6 1)。なお、主キー優先の場合、データソースの主キーのみで構成される分散インデックスが生成される。

【 0 1 0 1 】

5 0 7 では、分散インデックス作成部 1 2 3 が作成した分散インデックスを二次記憶装置 1 1 5 に格納し、5 0 8 で、分散インデックス情報 2 1 0 を更新 (なければ作成) を行う。特に、最終更新日付 2 1 5 を現在時刻に設定する。

10

20

30

40

50

【 0 1 0 2 】

一方、分散インデックス作成部 1 2 3 では、以下の処理を行う。6 0 1 で、5 0 6 で送られた分散インデックス管理部 1 1 4 からの要求を受取り、インデックス作成対象のデータソースの各レコードに対し、`getRecord()`を発行する(1 6 2)。得られたレコードのそれぞれから、インデックスカラムとキーカラムのユニオンとなるカラム集合を得て、結果の分散インデックスとして一時記憶領域に蓄積していく。そして、6 0 2 で、できあがった分散インデックスを分散インデックス管理部 1 1 4 に送る(1 6 3)。

【 0 1 0 3 】

以上が分散インデックス作成のインタフェースおよび処理手順である。

【 0 1 0 4 】

[6] 部分的な分散インデックスの作成

上述の手順では、分散インデックス作成部 1 2 3 は分散インデックス作成対象のデータソースの全レコードに対するインデックスを作成する。しかし、常に全レコードを対象にした分散インデックスを作成していると、データソースのデータ量が莫大である場合、分散インデックスのデータ量も大量となり、分散インデックスを保持するためのコスト、管理のためのコストが非常に大きくなる恐れがある場合がある。

【 0 1 0 5 】

このため本発明のシステムでは、分散インデックス作成のインタフェースのオプションとして、「分散インデックス作成条件」を分散インデックス作成時に用いる探索条件として管理アプリケーション 1 2 1 が指定できる。

【 0 1 0 6 】

分散インデックス作成時に、分散インデックス管理部 1 1 4 が分散インデックス作成条件を受取ると、前記 5 0 6 で、該分散インデックス作成条件をキーカラム、インデックスカラムとともに、分散インデックス作成対象のデータソースの分散インデックス作成部 1 2 3 に送る(1 6 1)。

【 0 1 0 7 】

該分散インデックス作成条件を受取った分散インデックス作成部 1 2 3 は、前記 6 0 1 で各レコードに対し、`getRecord()`を発行する(1 6 2)。得られたレコードのそれぞれに対し、該分散インデックス作成条件に合致するレコードのみを抽出し、インデックスカラムとキーカラムのユニオンとなるカラム集合を得て、結果の分散インデックスとして一時記憶領域に蓄積していく。この処理によって、結果としてできあがる分散インデックスのデータ量を、管理アプリケーション 1 2 1 の指定した分散インデックス作成条件にしたがって制御することが可能となる。

【 0 1 0 8 】

分散インデックス作成条件としては、例えば「住所= ' 東京 ' 」のような指定のほか、「全体の X % を選択」という条件を許す。「全体の X % を選択」が指定された場合、分散インデックス作成部 1 2 3 は `getRecord()` で得られたレコード群のうち、全体の X % を乱数発生により選択する。この方法により、データソースの全体傾向を統計的に分析するアプリケーション等、すべてのレコードに対するインデックスが必ずしも必要でない場合に好適な分散インデックスを作成することが可能となる。

【 0 1 0 9 】

[7] 分散インデックスの選択的な使用

分散インデックスはデータソース 1 0 7 への更新とは独立にデータ処理システム 1 0 0 が保持されるので、分散インデックスの内容とデータソース 1 0 7 中のデータとが一時的に不一致を生じる場合がある。このため、アプリケーションによっては、分散インデックスを選択的に利用して、最新データをアクセスする必要が生じる場合がある。また、前述のように「全体の X % を選択」という指定で作成した分散インデックスは、全体傾向を統計的に分析する等、特定のアプリケーションに特に合致するが、他のアプリケーションには不適な場合もある。

【 0 1 1 0 】

このため本発明のシステムでは、分散インデックスを選択的に使用方法をアプリケーション 120 に提供する。

【0111】

分散インデックスを探索的に使用する第1の方法として、分散インデックスの最終更新時刻等に関する探索条件を指定する方法を提供する。この方法では、問合せ発行前または問合せ発行時に、分散インデックスに対する探索条件を与えることによって、分散インデックスを選択する。例えば、「最終更新時刻が1週間以内である分散インデックスを使用許可」、「最終更新時刻が1週間以内で、対象データソースが取引履歴である分散インデックスを使用」等である。この指定は、前記ステップ303で、分散インデックスを選択する際に分散インデックス適用部112が評価し、条件に合致する分散インデックスのみを前記ステップ304以降で処理する。

10

【0112】

分散インデックスを選択的に使用する第2の方法として、分散インデックスの名称を明示的に指定する方法である。「インデックスID 211がIX11である分散インデックスの使用許可」等である。この指定も、前記ステップ303で、分散インデックスを選択する際に分散インデックス適用部112が評価し、条件に合致する分散インデックスのみを前記ステップ304以降で処理する。

【0113】

以上の処理により、各アプリケーションが分散インデックスを選択的に利用することが可能となる。

20

【0114】

【発明の効果】

(1) データソース107に対する分散インデックスをデータ処理システム100にあらかじめ生成、分散インデックス適用部112が分散インデックスを用いた問合せの変形と分解を行うことにより、レガシーAPや三次記憶などのデータソースに対する高速なアクセスが実現できる。

【0115】

(2) 分散インデックス作成部123をデータソース107に配置することにより、分散インデックス作成に際し、大量通信の発生を避ける。これにより、ネットワーク負荷が大幅に軽減される。また、ネットワーク負荷の軽減にともない、データソースを保持する計算機のネットワーク処理負荷も大幅に軽減される。

30

【0116】

(3) インデックス更新インタフェースをデータ処理システム100が提供し、インデックス更新要求を受け取ったら分散インデックス作成部123が分散インデックスを作成する。このインタフェースにより、適切なタイミングで分散インデックスの更新が実現される。また、分散インデックスを使うか使わないか、どれを使うかを指定するインタフェースを備えることにより、適切な分散インデックスを選択的に利用することが可能となる。

【0117】

(4) 分散インデックスとして、分散インデックス適用部112がデータソースの一部のレコードを対象とした分散インデックスを用いる。これにより、分散インデックスのデータ量を削減、大量のデータを保持するデータソースに対する分散インデックス作成が可能となる。

40

【0118】

以上4つの効果により、企業内、企業間の複数のDBMSを統合する情報基盤の統合に際し、リレーショナルデータベース管理システムに格納されたデータのみならず、レガシーAPや三次記憶等、問合せを効率的に実行できないデータソースに格納されたデータの統合が可能となり、これらデータソースに対する高速な問合せが実現できる。

【図面の簡単な説明】

【図1】実施例の全体構成を示すブロック図。

【図2】データ構造の構成図。

50

【図 3】分散インデックス適用の処理を示すフローチャート。

【図 4】分散インデックス利用を含む問合せ実行の処理を示すフローチャート。

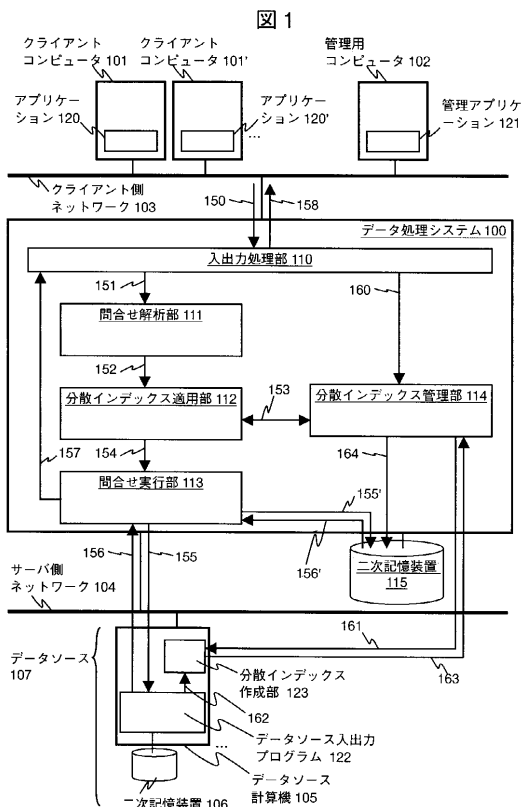
【図 5】分散インデックスの作成における分散インデックス管理部側の処理を示すフローチャート。

【図 6】分散インデックスの作成におけるインデックス作成プログラム側の処理を示すフローチャート。

【符号の説明】

100：データ処理システム
 101, 101' , ...：クライアントコンピュータ
 102：管理用コンピュータ
 103：クライアント側ネットワーク
 104：サーバ側ネットワーク
 105：データソース計算機
 106：二次記憶装置
 107：データソース
 110：入出力処理部
 111：問合せ解析部
 112：分散インデックス適用部
 113：問合せ実行部
 114：分散インデックス管理部
 115：二次記憶装置
 120, 120' , ...：アプリケーション
 121：管理アプリケーション
 122：データソース入出力プログラム
 123：分散インデックス作成部。

【図 1】



【図 2】

図 2

分散インデックス情報 210

インデックス ID 211	IX1
対象データソース 212	取引履歴
インデックスカラム 213	c1, c2, k1, k2, k3
キーカラム 214	k1, k2, k3
インデックス格納テーブル 215	T1
最終更新日時 216	2000 年 1 月 20 日 13:00

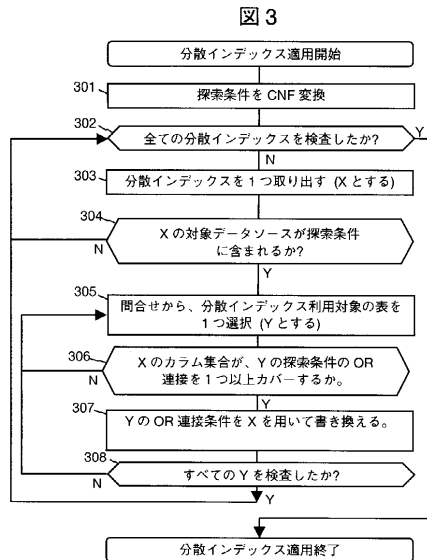
データソース情報 220

データソース名 221	取引履歴
主キー 222	(c1, c2, c3)
分割 223	(c1, c2)
内蔵インデックス 224	IX1(c1, c2), IX2(c1, c3), ...

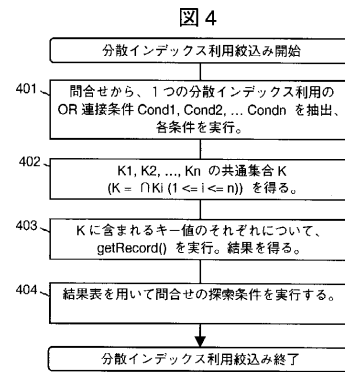
10

20

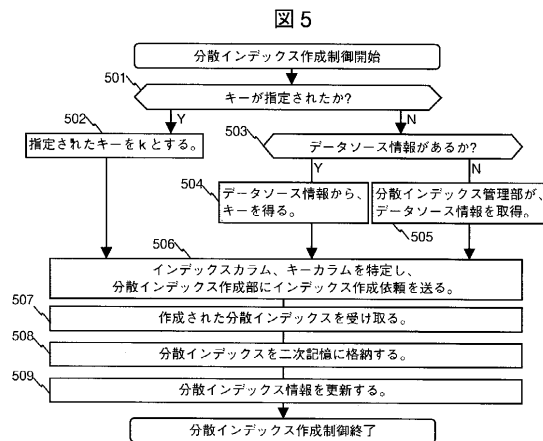
【図 3】



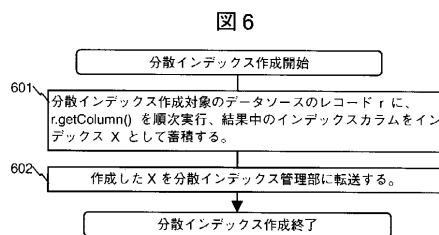
【図 4】



【図 5】



【図 6】



フロントページの続き

(72)発明者 清水 晃

東京都国分寺市東恋ヶ窪一丁目 2 8 0 番地

株式会社日立製作所中央研究所内

審査官 工藤 嘉晃

(56)参考文献 特開平 0 4 - 2 9 1 4 5 8 (J P , A)

特開平 0 5 - 1 8 9 2 8 5 (J P , A)

特開平 0 8 - 1 1 5 3 4 1 (J P , A)

特開平 1 1 - 0 1 5 8 4 5 (J P , A)

(58)調査した分野(Int.Cl. , D B 名)

G06F 12/00

G06F 17/30

JSTPlus(JDreamII)