(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号

特許第4955876号
(P4955876)

(45)発行日　平成24年6月20日(2012.6.20)　(24)登録日　平成24年3月23日(2012.3.23)

(51)Int.Cl.　F I
G06F 17/30 (2006.01)　G06F 17/30 340D
G06F 12/00 (2006.01)　G06F 17/30 180D
G06F 12/00 513D

請求項の数 21　(全 13 頁)

(21)出願番号　特願2001-296553 (P2001-296553)
(22)出願日　平成13年9月27日 (2001.9.27)
(65)公開番号　特開2002-163290 (P2002-163290A)
(43)公開日　平成14年6月7日 (2002.6.7)
審査請求日　平成20年9月24日 (2008.9.24)
(31)優先権主張番号　09/671458
(32)優先日　平成12年9月27日 (2000.9.27)
(33)優先権主張国　米国 (US)

(73)特許権者　500046438
マイクロソフト　コーポレーション
アメリカ合衆国　ワシントン州　９８０５２－６３９９　レッドモンド　ワン　マイクロソフト　ウェイ
(74)代理人　100140109
弁理士　小野　新次郎
(74)代理人　100089705
弁理士　社本　一夫
(74)代理人　100071124
弁理士　今井　庄亮
(74)代理人　100076691
弁理士　増井　忠弐
(74)代理人　100075270
弁理士　小林　泰

最終頁に続く

(54)【発明の名称】クエリ最適化のためのコストに基づく具体化ビューの選択

(57)【特許請求の範囲】
【請求項１】
データベース・クエリの実行に用いるための選択肢の拡張テーブルを提供する、コンピュータにより実行される方法であって、
前記コンピュータが、選択肢のテーブルを得るステップであって、前記選択肢のテーブルが、いくつかのグループを含み、１つのグループが、前記データベース・クエリを表すルート・エントリを有し、他のグループが、前記データベース・クエリの下位式を表すルート・エントリを有する、ステップと、
前記コンピュータが、前記クエリのための候補ビューを、複数の具体化ビューから選択するステップと、
各ルート・エントリに対し、
前記コンピュータが、前記ルート・エントリの演算子ツリーを抽出するステップと、
前記コンピュータが、各々の候補ビューに関し、該候補ビューのビュー定義から演算子ツリーを抽出するステップと、
前記コンピュータが、前記ルート・エントリおよび前記候補ビューの演算子ツリーを照合するステップと、
一致が得られた場合、前記コンピュータが、前記ルート・エントリの前記グループを対応する前記候補ビューで拡張することによって、対応する前記候補ビューを用いて前記選択肢テーブルを拡張するステップと、
を備えた方法。

【請求項２】
請求項１記載の方法において、ルート・エントリの前記演算子ツリーは二進ジョインを含み、前記コンピュータがルート・エントリの演算子ツリーを抽出するステップは、前記コンピュータが演算子をクエリ・グラフに隠すステップを含む、方法。
【請求項３】
請求項２記載の方法において、前記コンピュータが前記演算子をクエリ・グラフに隠すステップは、前記コンピュータが基礎テーブルを、これらに適用する述語と共にリストにまとめるステップを含む、方法。
【請求項４】
請求項１記載の方法において、前記コンピュータがビューの演算子ツリーを抽出するステップは、前記コンピュータが演算子を原始グラフ・ツリーに隠すステップを含む、方法。
【請求項５】
請求項４記載の方法において、原始グラフ・ツリーを、特定の演算子集合に制限し、更に原始テーブルに制限する、方法。
【請求項６】
請求項５記載の方法において、ジョイン・グラフのための前記演算子ツリーにおいて、ジョイン、外部ジョイン、およびフィルタ演算子のみを許可する、方法。
【請求項７】
請求項１記載の方法において、ルート演算子を追加することによって、前記選択肢テーブルを拡張する、方法。
【請求項８】
請求項１記載の方法であって、更に、
前記コンピュータが、コストに基づく最適化装置を用いて、前記選択肢の拡張テーブルに基づいて実行計画を選択するステップを含む、方法。
【請求項９】
請求項１記載の方法において、照合するステップは、更に、
各ルート・エントリに対し、
前記コンピュータが、前記具体化ビューの定義から前記演算子ツリーへの包含マッピングを試みるステップ、
を備えた方法。
【請求項１０】
請求項９記載の方法であって、更に、前記コンピュータが、前記包含マッピングの試みが成功した場合に残余演算を定義する、方法。
【請求項１１】
請求項１０記載の方法であって、更に、前記コンピュータが、前記選択肢テーブルに、定義した残余演算を追加するステップを含む、方法。
【請求項１２】
請求項１１記載の方法において、フィルタ、分類およびジョインから成るグループから、前記残余演算を選択する、方法。
【請求項１３】
請求項１から１２のいずれかに記載の方法をコンピュータに実行させる命令を有するコンピュータ読み取り可能記憶媒体。
【請求項１４】
データベース・クエリの実行に用いるための選択肢の拡張テーブルを提供するクエリ最適化装置であって、
選択肢のテーブルを得る手段であって、前記選択肢のテーブルが、いくつかのグループを含み、１つのグループが、前記データベース・クエリを表すルート・エントリを有し、他のグループが、前記データベース・クエリの下位式を表すルート・エントリを有する、手段と、

前記クエリのための候補ビューを、複数の具体化ビューから選択する手段と、
各ルート・エントリに対し、
前記ルート・エントリの演算子ツリーを抽出する手段と、
各々の候補ビューに関し、該候補ビューのビュー定義から演算子ツリーを抽出する手段と、
前記ルート・エントリおよび前記候補ビューの演算子ツリーを照合する手段と、
を備え、
一致が得られた場合、前記ルート・エントリの前記グループを対応する前記候補ビューで拡張することによって、対応する前記候補ビューを用いて前記選択肢テーブルを拡張するように構成された、
クエリ最適化装置。
【請求項１５】
請求項１４記載のクエリ最適化装置において、ルート・エントリの演算子ツリーは、二進ジョインおよび外部ジョインを含み、ルート・エントリの演算子ツリーを抽出する手段は、演算子をクエリ・グラフに隠す手段を含む、クエリ最適化装置。
【請求項１６】
請求項１５記載のクエリ最適化装置において、前記演算子をクエリ・グラフに隠す手段は、基礎テーブルを、これらに適用する述語と共にリストにまとめる手段を含む、クエリ最適化装置。
【請求項１７】
請求項１４記載のクエリ最適化装置において、ビューの演算子ツリーを抽出する手段は、演算子を原始グラフ・ツリーに隠す手段を含む、クエリ最適化装置。
【請求項１８】
請求項１７記載のクエリ最適化装置において、原始グラフ・ツリーを、特定の演算子集合に制限し、更に原始テーブルに制限する、クエリ最適化装置。
【請求項１９】
請求項１８記載のクエリ最適化装置において、ジョイン・グラフのための前記演算子ツリーにおいて、ジョイン、外部ジョイン、およびフィルタ演算子のみを許可する、クエリ最適化装置。
【請求項２０】
請求項１４記載のクエリ最適化装置において、ルート演算子を追加することによって、前記選択肢テーブルを拡張する、クエリ最適化装置。
【請求項２１】
請求項１４記載のクエリ最適化装置であって、更に、
コストに基づく最適化装置を用いて、前記選択肢の拡張テーブルに基づいて、実行計画を選択する手段を含む、クエリ最適化装置。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】
本発明は、一般的に、コンピュータの分野に関し、更に特定すれば、コストに基づいて具体化ビュー（materialized views）を選択するデータベース・クエリ最適化装置に関する。
【０００２】
【従来の技術】
この特許文書の開示内容の一部は、著作権の保護対象となる題材を含む。著作権者は、本特許文書または特許開示内容の、特許商標庁における特許ファイルまたは記録に現れる通りのファクシミリ再生については、いずれの者による場合であっても異議を唱えないが、それ以外についてはあらゆる著作権を保存することとする。以下の通知は、以下および図面に記載するソフトウエアおよびデータに適用されるものとする。著作権c２０００年Microsoft Corporation、版権所有。

【０００３】
リレーショナル・データベースは、データの行即ちタプル（tuple）の集合である。各行は、番号、名称、アドレス等のような情報を収容する１つ以上の列を有することができる。例えば、列は、従業員の名前、従業員ＩＤ、住所、電話番号、日毎の売上、およびその他の情報を含むことができる。この情報をデータベース内のテーブルに格納する。この特定のテーブルの行における情報は全て同じ人に関係付けられている。このようなクエリの１つを売上に関係付けることも可能である。このクエリは、ある日の各従業員毎の売上を見つけることに関係付けることができる。
【０００４】
具体化ビューは、１０年以上にわたってデータベース探索の主題であった。基本的なアイデアは、あるクエリの結果を具体化する、または格納し、同様のクエリがデータベースに提出されたときに、このように計算した結果を用いるということである。例えば、日毎の売上結果を、例えば、格納し、格納した結果を用いて、所与の月における振り上げ、または年間の総売上を含む、いくつかの関係するクエリに回答したいことがある。
【０００５】
【発明が解決しようとする課題】
柔軟性の最大化を図るためには、あるビューが存在する、または具体化されていることをアプリケーションが知る必要があるべきではない。クエリ・プロセッサは、ユーザ・クエリと既存の予備計算結果との間における一致を識別し、適用可能であればこのような結果を使うべきである。これは、ビュー利用問題として知られている。具体化ビューの集合体だけでなくベース・テーブル上にもユーザ・クエリを書き込んだ場合、どの具体化ビューを用いてこのようなクエリに答えることができるのか。次いで、どのビューを用いるか決定しなければならない。
【０００６】
トランザクションの正確性を保証するためには、ビューの内容は、ベース・テーブルの変更に対して同期して維持しなければならない。例えば、注文が入力または変更された場合、週毎の売上の具体化を更新し、変更を反映しなければならない。これは、ビュー保守問題として知られている。
【０００７】
具体化ビューは、これらがデータベースの物理的設計の一部であり、その主な目的が性能向上にある点で、インデックスと同様である。データベースの論理設計、およびアプリケーションの正確性は、具体化ビューの有無とは無関係でなければならない。インデックスと同様、具体化ビューは、クエリの性能に劇的な改善をもたらすことができる。
【０００８】
どのビューを用いるべきか決定する従来の試みでは、分離における問題を扱い、限定したシナリオを処理し、多くの場合クエリ全体をカバーする「グローバル」構造を想定していた。これは、「カバレッジの結果」（coverage results）を得るには有用である。例えば、この述語集合（set of predicates）およびこの形態のクエリを有するビューが与えられた場合、このアルゴリズムを用いて、クエリに答えるためにビューを用いることができるか否か決定する。任意のクエリを扱い、ビューの利用をクエリ最適化装置の実際のアーキテクチャ内に統合する必要がある。更に、一旦クエリに答えるためにビューが使用可能であることがわかったなら、それを用いるという問題に取り組む必要がある。
【０００９】
ビューの一致の目的のために、ユーザ・クエリに「グローバル」構造を構築することは、共通の最適化装置のアーキテクチャとは適合せず、「許されない」構造を用いた場合、不可能なこともある。ある複雑なクエリ上では、ビューの利用は、完全なクエリの下位式（sub-expression）においてのみ可能である。更に、これらサブクエリは、ある並び替えを行なった後にのみ現れる可能性があるが、これは当然選択肢の探査（exploration）プロセスにおいて行われる。
【００１０】

【課題を解決するための手段】
コストに基づくクエリ最適化装置は、具体化ビューのクエリに対する適用可能性を判定する。ビュー利用選択肢が最適化探査段階において生成するので、複雑なクエリにおける他の変換との相互作用が考慮される。具体化ビューを使用するか否かに関する最終判断は、推定コストに基づく。
【００１１】
最適化装置は、選択肢テーブルを生成し、クエリの各下位式に対する種々の可能性をコンパクトにエンコードする。演算子ツリーをテーブル内に暗示的に表現する。種々の可能性の探査中に、具体化ビューを検出および置換し、選択肢テーブルに追加する。具体化ビューおよび選択肢は、コストに基づくクエリ実行計画において用いるために選択する。
【００１２】
最適化装置は、コストを推定し、費用のかかる解決手段を間引きし、演算子ツリーを組み立て、最適な解決手段を構築するために用いられる。所与のクエリに対して、一般的な数の候補ビュー、およびビューの利用を考慮することができるある数のテーブル・エントリがある。クエリ内において参照されたテーブル、またはクエリが総計を含むか否かというような情報を用いて、関連があり得るビューを決定する。
【００１３】
考慮する演算子ツリーの数を削減するために、隠し（collapsed）演算子ツリーを生成し、基本的に全ての基礎テーブルを、それらに適用する述語と共にリストに纏めたクエリ・グラフを形成する。クエリ・グラフに一致するビューを、選択肢テーブルに追加する。また、選択肢テーブルから、原始グラフ・ツリーを抽出する。このような原始グラフ・ツリーは、特定の演算子集合のみを許可し、原始テーブルのみを許す。これによって、原始データベース・テーブルを優先して、ビュー・テーブルを無視することが可能になる。
【００１４】
２つの演算子ツリーが同一である必要はない。一方の演算子ツリーが他方の演算子ツリーを包含する場合、残余演算子を用いることができる。残余式は、フィルタを含み、演算子別にグループ化し、演算子を加入することができる。
【００１５】
【発明の実施の形態】
以下の本発明の実施形態の一例の詳細な説明では、その一部をなす添付図面を参照する。図面では、本発明を実施可能な具体的な実施形態の例を、例示として示す。これらの実施形態は、当業者が本発明を実施することができるように十分詳細に説明してあり、更に、他の実施形態も利用可能であり、論理的、数学的、電気的およびその他の変更も、本発明の精神または範囲から逸脱することなく行なえることは理解されよう。したがって、以下の詳細な説明は、限定的な意味で捕らえてはならず、本発明の範囲は添付した特許請求の範囲によってのみ定義されることとする。
【００１６】
詳細な説明は多数の章に分かれている。第１章は、本発明を実現するコンピュータ・システムの動作について説明する。続いて、どのようにして、コストに基づいてクエリ最適化装置による考慮のために、全体的な具体化ビューを識別し、選択肢テーブルに追加するかについて説明する。結論は、潜在的な利点をいくつか述べ、更に別の代替実施形態についても述べる。
ハードウエアおよび動作環境
本発明を実現するシステム例は、図１における計算デバイス１００のような計算デバイスを含む。その最も基本的な構成では、計算デバイス１００は、典型的に、少なくとも１つの演算装置１０２およびメモリ１０４を含む。計算デバイスの正確なコンフィギュレーションおよび形式に応じて、メモリ１０４は、揮発性（ＲＡＭのような）、不揮発性（ＲＯＭ、フラッシュ・メモリ等のような）、またはこれら２つの何らかの組み合わせとなる場合もある。この最も基本的なコンフィギュレーションを、図１における破線１０６で示す。

【００１７】
デバイス１００は、追加の機構／機能性を含むこともできる。例えば、デバイス１００は、磁気または光ディスクまたはテープを含むがこれらには限定されない、追加のストレージ（リムーバブルおよび／または非リムーバブル）を含むことができる。このような追加のストレージは、図１では、リムーバブル・ストレージ１０８および非リムーバブル・ストレージ１１０によって例示されている。コンピュータ記憶媒体は、揮発性および不揮発性、リムーバブルおよび非リムーバブル媒体を含み、コンピュータ読み取り可能命令、データ構造、プログラム・モジュールまたはその他のデータのように、情報格納技術のあらゆる方法で実現されている。メモリ１０４、リムーバブル・ストレージ１０８および非リムーバブル・ストレージ１１０は、全てコンピュータ記憶媒体の例である。コンピュータ記憶媒体は、ＲＡＭ、ＲＯＭ、ＥＥＰＲＯＭ、フラッシュ・メモリ、またはその他のメモリ技術、ＣＤ－ＲＯＭ、ディジタル・バーサタイル・ディスク（ＤＶＤ）またはその他の光ストレージ、磁気を用いたストレージ、あるいは所望の情報を格納するために使用可能であり、デバイス１００がアクセス可能なその他のあらゆる媒体を含むが、これらに限定される訳ではない。このようなコンピュータ記憶媒体はいずれも、デバイス１００の一部となり得る。

【００１８】
また、デバイス１００は、通信接続部１１２も内蔵し、当該デバイスは他のデバイスと通信することができる。通信媒体は、典型的に、読み取り可能命令、データ構造、プログラム・モジュール、あるいは搬送波またはその他の輸送機構のような変調データ信号におけるその他のデータを具体化し、あらゆる情報配信媒体を含む。「変調データ信号」という用語は、信号内に情報をエンコードするように変更された、１つ以上のその特性集合を有する信号を意味する。限定ではなく一例として、通信媒体は、有線ネットワークまたは直接配線接続のような有線媒体や、音響、ＲＦ、赤外線およびその他のワイヤレス媒体のようなワイヤレス媒体を含む。ここで用いる場合、コンピュータ読み取り可能媒体とは、記憶媒体および通信媒体双方を含むこととする。

【００１９】
また、デバイス１００は、キーボード、マウス、ペン、音声入力デバイス、接触入力デバイス等のような入力デバイス１１４を有することもできる。ディスプレイ、スピーカ、プリンタ等のような出力デバイス１１６も含ませることができる。これらのデバイスは、当技術では周知である。

【００２０】
本発明は、１つ以上のコンピュータまたはデバイス１１０のようなその他のデバイスによって実行する、プログラム・モジュールのようなコンピュータ実行可能命令のコンテクストにおいて説明するとよい。通常、プログラム・モジュールは、ルーチン、プログラム、オブジェクト、コンポーネント、データ構造等を含み、特定のタスクを実行するか、または特定の抽象的データ・タイプを実装する。典型的に、プログラム・モジュールの機能性は、種々の実施形態に必要に応じて組み合わせたりあるいは分散することができる。

【００２１】
オブジェクト指向プログラミング言語を含む多くの異なる方法を用いて、ソフトウエアを設計することができる。Ｃ＋＋およびＪａｖａ（登録商標）は、オブジェクト指向プログラミングに関連する機能性を与える、共通のオブジェクト指向コンピュータ・プログラミング言語の２つの例である。オブジェクト指向プログラミング方法は、データ・メンバ（変数）、および当該データ上で動作し、クラスと呼ばれる単一のエンティティを形成するメンバ関数（メソッド）をカプセル化する手段を与える。また、オブジェクト指向プログラミング方法は、既存のクラスに基づいて新たなクラスを作成する手段も与える。

【００２２】
オブジェクトとは、クラスのインスタンスである。オブジェクトのデータ・メンバは、コンピュータ・メモリ内部に格納されている属性であり、メソッドは、このデータ上で、潜在的に与えることができる他のサービスと共に作用する、実行可能なコンピュータ・コー

ドである。本発明では、発明の態様を一実施形態においてオブジェクトとして実現するというオブジェクトの観念を利用する。

【００２３】

インターフェースとは、名前のあるユニットに編成された、関連する関数のグループである。各インターフェースは、ある識別子によって一意に識別することができる。インターフェースはインスタンス化（instantiation）を有さない。即ち、インターフェースは、定義のみであり、当該インターフェースが指定するメソッドを実装するために実行可能コードを必要としない。オブジェクトは、インターフェースが指定するメソッドに実行可能コードを与えることによって、インターフェースを支援することができる。オブジェクトが供給する実行可能コードは、インターフェースが指定する定義を満たさなければならない。オブジェクトは追加のメソッドを与えることもできる。インターフェースは、オブジェクト指向プログラミング環境における、またはオブジェクト指向プログラミング環境による使用に限定される訳ではないことを、当業者は認めよう。

【００２４】

本発明は、機能ブロックを含むフローチャートを用いて説明する。ブロックは、必要に応じて、１つ以上のソフトウエアまたはハードウエア・モジュールで実現することができ、データベース・システムのコンテクストにおいて、計算デバイス１００上で実行する。

具体化ビューの識別

クエリ最適化装置は、通常、図２に示すように最初に簡略化段階があり、続いて、選択肢の探査および実行計画のコストに基づく選択がある。２１０において、元のクエリを識別する。簡略化／正規化段階２２０の間、元のクエリに対して、可能なときには、選択をプッシュ・ダウンする、あるいはサブクエリをジョイン（join）として書き直すというような何らかの変更を行なう。これらの変更は、「より良い」クエリ２３０を得ることを目的とする。典型的に、この段階では詳細なコスト推定がなく、前述の図における結果Ｑ’として、単一の「より良い」クエリ２３０を生成する。

【００２５】

探査段階２４０は、Ｑ’を受け取り、多数の選択肢を生成する。また、探査段階２４０は、詳細なコスト・モデルを用いて、最も安い推定実行コストを選択する。クエリ最適化装置は、最低のコストを有するクエリを実行する計画２５０を与える。

【００２６】

探査段階には、２つの標準的なアーキテクチャ、即ち、ボトムアップ・ダイナミック・プログラミング・ジョイン列挙、および変換によって促される選択肢の生成がある。双方のアーキテクチャは、選択肢テーブルを設定する。このテーブルは、クエリの各下位式毎に、種々の可能性をコンパクトにエンコードしたものである。

【００２７】

クエリ簡略化の間に具体化したビューを考慮することは可能であるが、これは不適当である。何故なら、単一の解決手段しか得ることができず、この選択を行なう詳細なコスト情報がないからである。また、あるビューの使用が、クエリの他の何らかの変換または修正が行われるまで、明白にならない場合もある。これが有効になるのは、クエリが具体化ビューの定義に非常に近づいたときのみであり、その場合でも、この特定のクエリに対して、ベース・テーブル内により良いインデックス集合があるなら、元のクエリの方がビューの使用よりも速い可能性もある。

【００２８】

この問題を解決するために、探査およびコストに基づく選択の間にビューの検出および置換を行なう。ここでは、これを変換に基づく最適化装置について記載し、一般的な原理は、ボトムアップ列挙を含む、他の選択肢テーブル構築方法にも及ぶ。

選択肢テーブルの増強

探査の間に具体化ビューを検討する場合、このような具体化ビューを用いるエントリによって、選択肢のテーブルを増強することから成る。元のクエリが、テーブルＡ，Ｂ，Ｃ上でのジョインであると仮定する。通常の選択肢テーブルは、図４の３００に概略的に示

すように見え、論理演算子のみである。選択肢テーブル３００は、４つのグループ３１０，３２０，３３０および３４０から成る。グループ３１０は、ＡＢＣのルート・エントリを有し、異なる選択肢を有するエントリを含む。グループ３２０，３３０および３４０は、それぞれ、ＡＢ，ＢＣおよびＡＣのルート・エントリを有する。尚、各々につき、ジョインの順序関係があるのは、２つの代替エントリのみである。

【００２９】

選択肢テーブルを見渡すことによって、図４において概略的に４００で示すような、エンコード演算子ツリーを得る。ルート・エントリ（上述のエントリにおけるＡＢＣ）から開始し、各エントリから演算子を選択する。各エントリにおいて最初の選択を行なうことによって、演算子ツリー４００を形成した。ＡおよびＢが最初に結合され、続いてその結果をＣと結合する。

【００３０】

具体化ビューＶ＝Ａ ｊｏｉｎ Ｂがある場合、具体化ビュー・テーブルＶｔが格納されており、これは、ＡおよびＢのジョインの結果を含む。これはジョイン下位式を得る有効な方法であるので、この選択肢によって選択肢テーブル３００を増強し、図５における増強テーブル５００を形成する。増強を５１０で識別し、ルート・エントリＡＢを有するグループに対するエントリとして追加する。

【００３１】

ここで生成することができ、最適化装置によって考慮する有効な演算子を図６の６００に示す。これは、ＶｔおよびＣのジョインから成る。選択肢テーブルを増強する正確な機構は、最適化装置のアーキテクチャによって異なる。変換に基づく最適化装置の場合、新たな変換規則をシステムに追加することによって、拡張が得られる。ボトムアップ・ジョイン列挙では、構造手順を変更する必要がある。一旦選択肢をテーブルに追加したなら、コストを推定し、費用のかかる解決手段を間引き、演算子ツリーを組み立て、最適な解決手段を構築する通常の最適化装置の機構を適用する。

【００３２】

特定のクエリに対し、通常、ある数の候補ビューＶ１，Ｖ２，…Ｖｋ、およびビューの利用を考慮することができるある数のテーブル・エントリがある。クエリに関係がある可能性があるビューのみを考慮すればよい。クエリにおいてどのデータベース・テーブルを参照するか、およびクエリが総計を含むか否かについての情報を用いることにより、ビューを関連なしとして識別する。他の情報も使用可能である。これによって、候補のビュー集合を狭めることができる。各テーブル・エントリ上で同様の情報を用い、無関連として検出され得るビュー定義を一致させようとするのを回避することができる。

【００３３】

（ビュー、テーブル・エントリ）の対を多数考慮するために、テーブル・エントリを固定し、次いで多数の候補ビューを用いて照合を試みる。これによって、動作を開始する前に、所与のテーブル・エントリにおいて全ての追加選択肢を生成する。この順序は、通常の最適化の順序と一致しており、各エントリ毎に単一の一致構造を生成し、それを各候補ビュー毎に再利用することを可能にする。

【００３４】

特定の具体化ビューＶ、および適用可能性を検査しようとする特定のテーブル・エントリＥが与えられると、図７のフローチャートに示すように以下のステップを実行する。７１０において、エントリＥのために演算子ツリーＴを抽出する（このステップは、同じエントリＥ上の種々のビューに対し共有する）。次に、７２０において、ビュー定義Ｖから演算子ツリーＴへの包含マッピングを試みる。マッピングに成功すると、残余演算が行われ、７３０においてＴ＝Ｏｐ（Ｖｔ）となる。７５０においてビューの利用により、選択肢テーブルを拡張する。７４０において、残余演算が新たなテーブル・エントリの導入を必要とする場合もある。これらのステップの更に詳細な説明を以下に行なう。

照合のための演算子ツリーの抽出

テーブル・エントリは多数の演算子ツリーに対応する。所与のエントリからエンコードさ

れた各演算子ツリーを抽出し考慮することは、これらの数が指数的に増大すると（関与するテーブル上で）、実現不可能となる。非決定論的に単一の演算子ツリーを抽出することは不適当である。何故なら、ビュー定義に一致しない場合も起こり得るからであるが、なおも他のあるエンコード演算子ツリーが定義と一致する可能性もある。

【００３５】

この問題の一例は、Ａ　ｊｏｉｎ　ＢおよびＢ　ｊｏｉｎ　Ａ双方をエンコードするグループについて考慮する場合の、Ａ　ｊｏｉｎ　Ｂの形態のビュー定義である。厳格な演算子マッピングを行なえば、２つのエンコード・ツリーの内１つだけに継承される。あるいは、全ての演算子ツリーを素早く抽出すると、実現不可能になる。２つのテーブルには２つの演算子ツリーがあり、３つのテーブルには１２個、４つのテーブルには既に１２０の演算子ツリーがある。

【００３６】

この問題に取り組むために、相補的な技法を用いる。第１の技法では、照合は、二進ジョイン（binary join）を含む演算子ツリーに対して行なわず、クエリ・グラフとして知られている、演算子を隠した形態に対して行なう。これは、基本的に、全ての基礎テーブルを、これらに適用される述語と共にリストに纏めたものである。一旦演算子ツリーを抽出したなら、ジョインを隠し、次いでクエリ・グラフを比較する。この方法は、演算子ツリーＡｊｏｉｎ　ＢまたはＢ　ｊｏｉｎＡのいずれかを抽出し、具体化ビューＡ　ｊｏｉｎ　Ｂとの一致という同じ結果が得られる。クエリ・グラフは、少なくともジョインおよび外部ジョイン（outerjoin）を表わすことができることが知られている。

【００３７】

第２の技法では、選択肢テーブルから原始グラフ・ツリー（primitive graph-tree）を抽出し、ビューと照合する。このツリーは、特定の演算子集合のみを表出させることができる。ジョイン－グラフでは、抽出したツリーの中においてジョインおよびフィルタ演算子のみが許される。条件が長い値のリストを有するＩＮリストである場合、あるフィルタ条件を半ジョイン（semijoin）に転換することが可能であるが、半ジョインを含む式はクエリ・グラフに直接マッピングしないものの、フィルタを有する式はマッピングする。したがって、望ましいのは、半ジョイン・ツリーではなく、フィルタ・ツリーである。同様に、場合によっては、ＯＲ条件がＵＮＩＯＮに転換する。ジョイン－グラフ抽出では、テーブル・エントリのために演算子ツリーを組み立てるとき、ジョインのみが有効な演算子と見なされる。

【００３８】

また、グラフ・ツリーは、原始テーブルのみを表出させることも可能である。テーブル・エントリは、既にビューへの参照を含む場合もあるが、これらは特定のエントリのツリーの組み立てでは、除外すべきである。先に考慮した３つのテーブルのジョインでは、演算子ツリーＶｔｊｏｉｎ　Ｃの抽出は、定義がＡ　ｊｏｉｎ　Ｂ　ｊｏｉｎ　Ｃであるビューと一致しない。したがって、原始データベース・テーブルを優先して、ビュー・テーブルは無視される。

【００３９】

前述のように、所望の形態を有する演算子ツリーを抽出し、所与のテーブル・エントリに対して、そのクエリ・グラフを１回だけ構築する。次いで、得られたクエリ・グラフを再利用して、多数の候補ビューを照合する。

【００４０】

一実施形態では、２つの広い式クラス、グラフ・ツリーまたはグラフ・ツリー上のＧｒｏｕｐ　Ｂｙ（分類）のいずれかを考慮する。更に別の実施形態では、他のクラスも考慮することができる。特定のテーブル・エントリに対して、２つのツリーを、２つのビュー・クラスについて抽出する試みを行なう。

包含マッピング

ビュー定義を抽出したツリーと比較する場合、テーブル参照、述語、およびこれら２つの間の他のスカラー式の間で、マッピングを設定する。しかし、２つの式は、同一である必

要はなく、多くの場合同一でない。その理由は、通常の探査プロセスは、クエリに対して可能な同等の演算子ツリーを全て考慮する訳ではないからである。例えば、３つの演算子ツリー８１０，８２０および８３０を図８に示す。第１演算子ツリー８１０は、ビュー定義であり、第２演算子ツリー８２０は、ビューがサブツリー上で直接一致する場合のクエリである。しかしながら、この第２演算子ツリー８２０は、初期選択評価を考慮するだけの最適化装置の通常の探索空間では決して考慮されない（何故なら、それらの後の評価は無駄であるからである）。次に、同一でない、第３演算子ツリー８３０とビューを照合しなければならない。

【００４１】

この「正確でない」一致は、最適化装置の探索空間に対する制限の結果であり、処理すべき類似度は、最適化装置によって実施される探索空間を考慮する必要がある。この相違のために、残余演算子が必要となる。前述の場合では、ビューの照合によって、残余演算子ツリー８４０を生成する。残余式は、前述の場合に示したように、フィルタ、ＧｒｏｕｐＢｙ（分類）、およびジョインを含むことができる。ＧｒｏｕｐＢｙは、「より低い粒度」を計算するために「より高い粒度」総計を用いる場合に用いられる。例えば、（領域、月）によって総売上を計算する既存の具体化ビューを用いて、領域別に（全ての月）の全売上を計算することができる。

【００４２】

例えば、ジョインは、より多くの列を得るために用いられる。例えば、具体化ビューが多数のジョインの結果を含む場合、これは顧客キーを格納するが、名前は格納しない。名前も要求するクエリでは、ビューを用いることができるが、残余演算は、既知のキーから名前を検索するために、顧客テーブルとの追加ジョインを含む。

【００４３】

ジョインまたは総計のための特定の残余演算子のいくつかの導出は、当業者には公知である。また、例えば、ＢおよびＣ間（またはＡおよびＣ間）に外部キーの制約がないのであれば、Ａｊｏｉｎ Ｂ ｊｏｉｎ Ｃという形態の具体化ビューを用いて、単一のジョインＡｊｏｉｎ Ｂを有するクエリに答えることができる。これらの技法は、本発明によって利用し、影響力を及ぼすことができる。実際には、これは、最適化装置全体の機構が、特定の形態のビューを用いてあるクエリに答えることができるか否かに関する既存の結果にリンクする点である。

選択肢テーブルの拡張

一旦具体化ビュー選択肢を求めたなら（ビュー参照および恐らくは残余演算子）、これを選択肢テーブルに追加しなければならない。ＳＱＬサーバが用いるような、変換に基づく最適化装置のコンテクストでは、これは、最適化装置フレームワークによって処理され、式を取り込み、ルート演算子を、考慮対象の元のテーブル・エントリに追加し、必要であれば、新たなテーブル・エントリを作成する。ボトムアップ列挙手法では、選択肢をテーブル・エントリに添付する汎用的な標準機構はないので、既存のユーティリティおよびシステムのその他の実施態様の詳細を考慮に入れて、これをコード化する必要がある。

【００４４】

一実施形態では、これまでの説明にしたがって、データベース・サーバの変換に基づく最適化装置に、以下の変更および追加を行なう。

・変換規則のためのパターンを記述するために用いる構造に、ＣＧｒａｐｈプリミティブを追加する。

【００４５】

・２つの変換規則を追加する。一方は形態グラフ・ツリーの具体化ビューを扱い、他方はグラフ・ツリー上で形態Ｇｒｏｕｐ　Ｂｙ（分類）の具体化ビューを扱う。

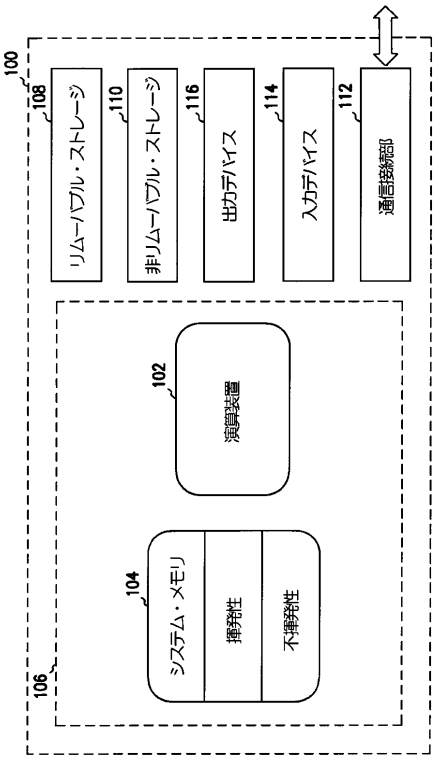
【００４６】

・二進ジョインを含む演算子ツリーを取り込み、クエリ・グラフを生成する関数を追加する。

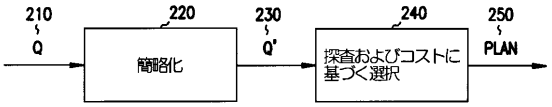
・２つのクエリ・グラフの包含照合を行い、照合に成功した場合、残余演算子を生成す

る関数を追加する。
【００４７】
・別々に発生するために用いた列を生成する新たなテーブルの円滑な集積のために、列マッピングを実行する関数を追加する。
結論
本願は、本発明のあらゆる改作および変形をも包含しようとするものである。本発明は、特許請求の範囲およびその均等物によってのみ限定されることを意図しているのは明白である。コストに基づくクエリ最適化装置によって、具体化ビューのクエリに対する適応可能性を決定することを可能にするシステムおよび方法について説明した。最適化の探査段階において、ビュー利用選択肢を生成するので、複雑なクエリにおける他の変換との相互作用が考慮に入れられる。具体化ビューを用いるか否かに最終判断は、推定したコストに基づいて行われる。
【図面の簡単な説明】
【図１】本発明を実現可能なコンピュータ・システムのブロック図である。
【図２】本発明によるクエリ最適化装置のブロック図である。
【図３】最適化の間に考慮する代替エントリのテーブルである。
【図４】図３のテーブルにおける１つエントリから形成した演算子ツリーの図である。
【図５】代替エントリの増強テーブルである。
【図６】具体化ビューを内部に組み込んだ演算子ツリーである。
【図７】代替エントリのテーブルに追加するビューを識別するプロセスを示すフローチャートである。
【図８】残余演算子ツリーを含む、多数の代替演算子ツリーの図である。
【符号の説明】
１００　計算デバイス
１０２　演算装置
１０４　メモリ
１０６　基本的コンフィギュレーション
１０８　リムーバブル・ストレージ
１１０　非リムーバブル・ストレージ
１１２　通信接続部
１１４　入力デバイス
１１６　出力デバイス
２１０　元のクエリの識別
２２０　簡略化／正規化段階
２３０　「より良い」クエリ
２４０　探査段階
２５０　計画
３００　選択肢テーブル
３１０，３２０，３３０，３４０　グループ
４００　エンコード演算子ツリー
５００　増強テーブル
５１０　増強を
６００　有効な演算子
８１０，８２０，８３０，８４０　演算子ツリー

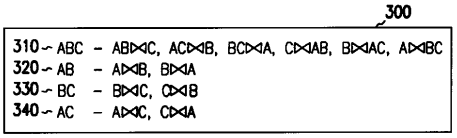
【図１】



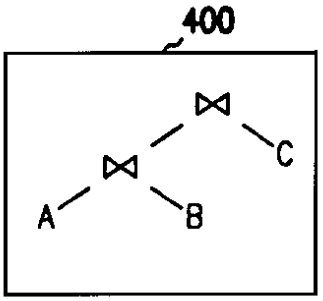
【図２】



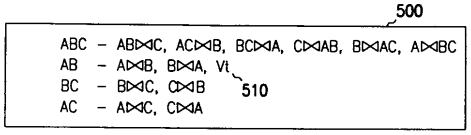
【図３】



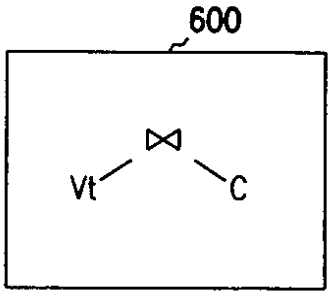
【図４】



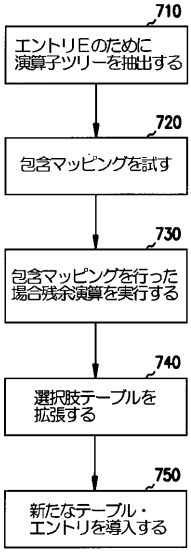
【図５】



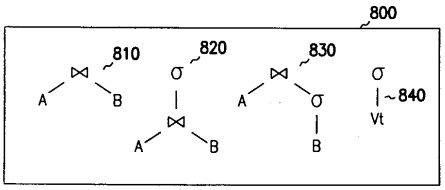
【図６】



【図７】



【図８】



フロントページの続き

(74)代理人 100096013
弁理士 富田 博行
(72)発明者 シーザー・エイ・ガリンドーレガリア
アメリカ合衆国ワシントン州９８０５２，レッドモンド，ノースイースト・フォーティフォース・コート １６１４１
(72)発明者 ミリンド・エム・ジョシ
アメリカ合衆国ワシントン州９８００７，ベルビュー，ノースイースト，ワンハンドレッドアンドフォーティエイトス・アベニュー ４７６３，アパートメント ピー２０２

審査官 野崎 大進

(56)参考文献 特表２００３－５２７６４２（ＪＰ，Ａ）
特表２００２－５１００８８（ＪＰ，Ａ）
米国特許第０５８９７６３２（ＵＳ，Ａ）
米国特許第０６０２６３９０（ＵＳ，Ａ）
国際公開第９９／０５０７３２（WO，Ａ１）
国際公開第９９／０５０７６２（WO，Ａ１）
特開平０５－２３３７２１（ＪＰ，Ａ）
特開平０７－３３４５２９（ＪＰ，Ａ）

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