

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号  
特許第4425355号  
(P4425355)

(45) 発行日 平成22年3月3日(2010.3.3)

(24) 登録日 平成21年12月18日(2009.12.18)

(51) Int.Cl.  
G06F 17/30 (2006.01)

F I  
G O 6 F 17/30 4 1 4 Z

請求項の数 7 (全 17 頁)

(21) 出願番号	特願平11-502711	(73) 特許権者	500046438
(86) (22) 出願日	平成10年6月1日(1998.6.1)		マイクロソフト コーポレーション
(65) 公表番号	特表2002-511169 (P2002-511169A)		アメリカ合衆国 ワシントン州 9805
(43) 公表日	平成14年4月9日(2002.4.9)		2-6399 レッドモンド ワン マイ
(86) 国際出願番号	PCT/US1998/011199		クロソフト ウェイ
(87) 国際公開番号	W01998/057278	(74) 復代理人	100115624
(87) 国際公開日	平成10年12月17日(1998.12.17)		弁理士 濱中 淳宏
審査請求日	平成17年6月1日(2005.6.1)	(74) 復代理人	100136490
(31) 優先権主張番号	08/871,079		弁理士 中西 英一
(32) 優先日	平成9年6月9日(1997.6.9)	(74) 代理人	100077481
(33) 優先権主張国	米国 (US)		弁理士 谷 義一
		(74) 代理人	100088915
			弁理士 阿部 和夫

最終頁に続く

(54) 【発明の名称】 データベース照会システムおよび方法

(57) 【特許請求の範囲】

【請求項 1】

コンピュータが供給されたサーチ値に基づいてデータベース内のデータを見つける方法であって、前記データベースがキー値に基づいてサーチすることができる複数のレコードを有し、前記方法が、  
下側キー値および上側キー値を含む第1および第2のキー値対を有する複数のページ定義エントリを参照することにより、前記複数のページ定義エントリの中から、前記供給されたサーチ値に対応する、前記供給されたサーチ値を前記第1のキー値対で規定される範囲に含む1つの前記ページ定義エントリを識別するステップと、  
サーチ・パラメータとして前記第2のキー値対を利用して前記データベースをサーチすることにより、前記第2のキー値対により規定される範囲に含まれるレコードを有する1つのデータベース・レコード集合を見つけるステップと、  
を含み、前記各ページ定義エントリの第2のキー値対が、1つの特定のデータベース・レコード集合の下側レコードおよび上側レコードと同値の前記下側キー値および上側キー値を含み、第1のページ定義エントリの前記第1のキー値対の下側キー値が最小値であり、第2のページ定義エントリの前記第1のキー値対の下側キー値と上側キー値が、最大値と1つの所与の前記ページ定義エントリの上側キー値であり、前記第2のページ定義エントリを除く前記複数のページ定義エントリの内、1つの所与の前記ページ定義エントリの第1のキー値対の前記上側キー値が異なった1つの前記ページ定義エントリの第1のキー値対の前記下側キー値と同値であることを特徴とする方法。

10

20

**【請求項 2】**

請求項 1 記載の方法において、

前記第 2 のキー値対が規定する範囲は、前記第 1 のキー値対が規定する範囲を含み、前記第 1 のキー値対が規定する範囲よりも広いことを特徴とする方法。

**【請求項 3】**

1 つのサーバと複数のクライアントを有するコンピュータ・システムにおいて、キー値に基づいてサーチすることができる複数のレコードを有するサーバ保持のデータベース内のデータを見つける方法であって、

下側キー値および上側キー値を含むインデックス・キー値対とバウンド・キー値対とを有する複数のページ定義を、前記サーバにおいて保持するステップと、

サーチ値を前記サーバからクライアントに提出するステップと、

前記複数のページ定義の内の、前記サーチ値を囲む前記インデックス・キー値対を有する 1 つの前記ページ定義を識別するステップと、

前記識別した 1 つのページ定義の前記バウンド・キー値対で前記データベース・レコードをサーチすることにより、前記識別した 1 つのページ定義の前記バウンド・キー値対により規定される範囲に含まれるレコードを有する 1 つのデータベース・レコード集合を識別するステップと、

前記識別したデータベース・レコード集合のデータを前記クライアントに返すステップと、

、  
を含み、第 1 のページ定義エントリの前記インデックス・キー値対の下側キー値が最小値であり、第 2 のページ定義エントリの前記インデックス・キー値対の下側キー値と上側キー値が最大値と 1 つの所与の前記ページ定義エントリの前記インデックス・キー値対の上側キー値であり、前記第 2 のページ定義エントリを除く前記複数のページ定義エントリの内、所与の 1 つの前記ページ定義の前記インデックス・キー値対の前記上側キー値は、異なる 1 つの前記ページ定義の前記インデックス・キー値対の前記下側キー値と同値であることを特徴とする方法。

**【請求項 4】**

請求項 3 記載の方法において、各ページ定義は、前記ページ定義が有する前記インデックス・キー値対の下側キー値および上側キー値と同一の下側キー値および上側キー値を含むバウンド・キー値対を有すること、を特徴とする方法。

**【請求項 5】**

請求項 3 記載の方法において、各ページ定義は、前記ページ定義が有する前記インデックス・キー値対が規定する範囲を含む前記バウンド・キー値対を有すること、を特徴とする方法。

**【請求項 6】**

請求項 3 記載の方法であって、さらに、前記クライアントに対し、前記識別したデータベース・レコード集合からの前記データと共に、前記インデックス・キー値対を返すステップを含むこと、を特徴とする方法。

**【請求項 7】**

請求項 3 記載の方法であって、さらに、前記クライアントに対し、前記識別したデータベース・レコード集合からの前記データと共に、前記バウンド・キー値対を返すステップを含むこと、を特徴とする方法。

**【発明の詳細な説明】****技術分野**

本発明は、データベースをリモートのクライアントを使用してサーチしブラウズする方法に関するものである。

**発明の背景**

大きなデータベース・アプリケーションは、通常は、あるフィーチャをサポートしていて、これによりユーザは、このユーザがサーチして求めようとしている実際の値にわずかに近似するに過ぎないサーチ・キーをエンターできるようにしている。しばしば、これは、

10

20

30

40

50

“ ”のような“ワイルドカード”キャラクタを使って行われている。この“ ”キャラクタは、可能性のある全てのキャラクタ・ストリングを表す。このため、“jon\*”に対するサーチは、ストリング“jon”で始まりそして任意の他のストリングのキャラクタが続くあらゆる値を検索することになる。“Jon”、“Jones”、“Jonathan”は全て、“jon\*”をサーチ・キーとして使用した結果として検索されることになる。

このサーチ方法の利点は、フレキシブルであり、ユーザに対し説明が容易であり、また実施が容易である点である。しかし、この方法は、幾分限定されたものである。例えば、あるサーチから返ってきた値がサーチしている値を含んでいない場合、この質問(query)を拡張または変更して“手近な”値を見つけ出すための明解な方法がない。さらに、この方法は、ユーザが知らないであるいは故意に、過大に広い質問、データベースの受け入れられない程大きな部分が返されるような広い質問をするのを許容する。例えば、あるユーザが“a”と質問をタイプするとすると、これは、“a”で始まる全ての値が返ってくることになる。

10

今日、データベースは、インターネットを介して質問を受けることが増大してきており、サーチ結果をユーザに対し比較的小さな部分で返すこともより重要となってきた。サービス・プロバイダの中には、計測した長さでサーチ結果を提供するプロバイダがある。例えば、サーチの最初の20個のヒットのみを返し、そしてユーザには、次の20個のヒットを要求するオプションを与える。しかし、このようなサービス・プロバイダは、その次の20個のヒットが要求されたときに、全く新たなサーチを開始し、そしてこの結果から第2のグループの20個のヒットを選択するようにしていることが多い。この理由は、サーバにおいて、あるデータベースを任意の所与の時刻にアクセスすることのある数千ものユーザに関して、状態情報を保持することが実際的ではないからである。このため、多くのソートされた行を返す質問は、サーバに対する各要求に関して完全に実行されることになる。これは、データをサーチし、ソートしそしてクライアントに伝送するため、サーバ・リソースのかなりの量を浪費する。

20

この従来技術は、データベース・ルックアップ技法を含み、この技法は、これらの問題にある程度対処している。この従来技術の技法によれば、2つのテーブルが、実際のデータベースに加えて保持されていた。第1のテーブルは、複数の行を有し、その各行がデータ・レコードの個々の集合に対応していた。各行は、3つの値、すなわち、最初(first)の値、最後(last)の値、そしてIDに対するフィールドを含んでいた。この最初と最後の値は、電話帳または辞書のトップにあるインデックス・エントリに類似のデータの集合またはページを定義していた。ある特定のテーブル・エントリに対応する1つのデータ・ページは、最初の値と最後の値との間の範囲内のデータ値を全て引くくめて含むことになる。このデータ・ページは、予め生成した別のページ内に置かれた。

30

1つの例として、以下のラスト・ネームのデータベースについて考えることにする。

Ableson Ahlquist Applegate Ashley Bachnlann Bailey	
Bangsberg Barker Barnes Baxter Beaver Becker Beckett	
Bergquist Beutel Bezanson Blackwell Boldman Borgman	
Bowman Bratsanos Braum Brown Burghardt Butler Cantwell	
Carlstrom Carolan Caudill Chen Cheng Chung Conway	
Crawford Cunneely Cutler Davidson Davies Davis Day De	
Donato DeBragga DeBroeck DeNike Dosch Douglas Doyle	10
Drage Driscoll Dugan Dulfer Duncan Dunkelberger Dunn	
Durham Edwards Elder Ellis Evans Ewing Fang Fay Ferguson	
Ferris Feny Fine Fitzgibbons Flippin Flowers Fort Francoeur	
Freeman Funaro Galasyn Garrity Gilroy Goodman Gordon	
Gower Grady Gui Guo Gwertzman Harbin Harkness Hart Hartill	
Haygood Hefiin Helfrich Hersey Hogan Honeycutt Homstein	20
Hunter Huse Jackson Jewiss Jones Joyner Kearney Kelly	
Kelnhofner King Kinsella Kramer Kreider Labyak Lalonde	
Laurie Lee Leptich Lim Lindell Livingston Loggins Lorenzana	
Lowe Ludden Lutz Lyons Maccalman Maes Mah Mahoney	
Malugen Martin Mastan Maumas Maxey Mc Mahon McCann	
McCrory McDaniel McGuire Mercure Meyers Miao Milford	
Miller Moffitt Monberg Moncure Monroe Mooney Moore	30
Morrison Murray Naroski Newton O'Connor O'Sullivan Oker	
Packham Perabo Peters Phan Pipkins Plamondon Prekeges Prets	
Quach Raines Ranch Rather Reid Reynolds Rice Richardson	
Rivera Roberts Robinson Rodrigues Rutledge Saunders	
Shamblin Simmons Simpson Slonsky Spahn St. Clair Stulz	
Sturm Sturms Teply Theivagt Thomson Tierney Timoney Titus	40
Troupe Tucker Tutt Twyman Tyner Utzschneider Vaile	
Visintainer Walker Ward Watson Webb Webb Whitaker White	
Wilcher Williams Wilson Wolff Yenne Younkin	

これに対応するテーブル（ページ定義テーブル（page definition table）と呼ぶ）は、以下のテーブル 1 に示すように現れることになる。

最初 (FIRST)	最後 (LAST)	ID
Ableson	Baxter	0
Beaver	Bowman	1
Bratsanos	Chen	2
Cheng	Day	3
De Donato	Dugan	4
Dulfer	Ewing	5
Fang	Fort	6
Francoeur	Grady	7
Gui	Helfrich	8
Hersey	Joyner	9
Kearney	Laurie	10
Lee	Lutz	11
Lyons	Maxey	12
McMahon	Miller	13
Moffit	Newton	14
O ' Conner	Prekeges	15
Prets	Rivers	16
Roberts	Spahn	17
St. Clair	Titus	18
Troupe	Ward	19
Watson	Younkin	20

10

20

#### テーブル 1

このテーブルによれば、ID 0 ( ページ 0 ) を有する情報ページは、“ Ableson ” で始まりそして “ Baxter ” で終わる 10 個のネームを含んでいる。ページ 1 は、“ Beaver ” で始まりそして “ Bowman ” で終わる次の 10 個のネームを有する。このページ定義テーブルは、各ページがほぼ等しい数のネームを有するように作成されていた。

第 2 のテーブルも保持されていた。このテーブルは、ID 番号で配列した、データベースからの実際のデータを有していた。上記例では、このようなテーブル ( ページ・データ・テーブル ( page data table ) と呼ぶ ) は、以下のように現れることになる。

30

ID	PAGEDATA
0	Ableson . . . Baxter
1	Beaver . . . Bowman
2	Bratsanos . . . Chen
3	Cheng . . . Day
4	De Donato . . . Dugan
5	Dulfcr . . . Ewing
6	Fang . . . Fort
7	Francoeur . . . Grady
8	Gui . . . Helfrich
9	Hersey . . . Joyner
10	Kearney . . . Laurie
11	Lee . . . Lutz
12	Lyons . . . Maxey
13	McMahon . . . Miller
14	Moffit . . . Newton
15	O'Conner . . . Prekeges
16	Prets . . . Rivers
17	Roberts . . . Spahn
18	St. Clair . . . Titus
19	Troupe . . . Ward
20	Watson . . . Younkin

10

20

テーブル 2

このテーブルの ID 値は、ページ定義テーブル（テーブル 1）の ID 値に対応している。PAGEDATA 値は、データベースから取った、特定のページに関しユーザに返すべき実際のデータから成っている。ここでの例においては、このデータは、単にネーム自体から成っている。実際のアプリケーションにおいては、電話番号や住所のような他のデータも含まれることがある。

この技法の下でサーチを実行するには、ユーザは、ネームの開始部分、あるいはこのネームに近似しているとユーザが考える何等かのストリングをエンターする。例えば、ユーザがサーチ・ストリングとして“file”をエンターするとする。このストリングは、テーブル 1 を質問するのに使用されて、このサーチ・ストリングに渡る（span）行の ID を見つけ出す。SQL フォーマットを使用すると、このサーチは、以下ようになる。

30

```
select ID from page_definition_table where
```

```
FIRST <= 'file' and LAST >= 'file'
```

ここで、“page\_definition\_table”はテーブル 1 を指している。

この場合、6 がその適当な ID であり、これは、ページ 6 をユーザに返すべきことを示す。その適当なデータを見つけ出すには、第 1 のサーチで識別した ID 番号を使用することにより、テーブル 2 に対し第 2 の質問を実行する。これを達成する SQL 質問は、以下の通りとなる。

40

```
select PAGEDATA from page_data_table where ID == X
```

ここで、“page\_data\_table”は、テーブル 2 を指し、X は第 1 の質問から返された ID 値である。PAGEDATA 値は、その ID 値と共にユーザに返されていたものである。この ID 値を使用することにより、ユーザは、テーブル 2 に対する質問を減分したあるいは増分した ID を使用して行うことにより、前のあるいは後のページをブラウズすることができていた。

この技法には、いくつかの深刻な問題がある。1 つの問題は、ユーザが、ページ定義テーブルの互いに隣接する行の最初の値と最後の値との間に入るサーチ・ストリングを提出したときに起きていた。例えば、“grant”のサーチ・ストリングは、テーブル 1 内に示したページ定義テーブルの行 7 と行 8 との間に入ることになる。これは、このサーチを失敗

50

とさせ、したがってデータが全くユーザに返されないことになる。

別の問題は、新たなデータのどのような挿入も、それらテーブルの両方の再生成を必要とすることであった。さらにまた、テーブル2のその構造のため、そのテーブルの生成の前にPAGEDATAフィールドを定義しなければならなかった。あるサーチから返されたこれらフィールドを変化させることは、PAGEDATAテーブル全体の再生成を必要としていた。これに  
10 関係する1つの問題は、そのような生成が、任意のサイズのデータベースを取り扱うときに、非常にかんりの量の時間を必要としたことである。サンプルのアプリケーションにおいては、再生成は、6時間以上の計算時間を必要としていた。

この従来技術の技法の別の問題は、ユーザ指定のサーチ・ストリング（これは、選択しそしてユーザに対し強調する）に対する“最も近い”一致が、サーチ結果内において、返されたデータの初めから予測できないオフセット場所にて現れていたことである。使い易さを向上させるには、このデータを、サーチ結果の初めかあるいはサーチ結果のおよそ中間の所のようなもっと予測できる場所に位置付けるのがより望ましい。

本願発明者は、上記の従来技術の方法に対する改良法を見い出したのであって、この改良法により、ユーザは、質問を提出し、そして小グループのデータを、最も近い一致が返された結果内のおよそ中央となったソートされたフォーマットで受けることができる。また、本発明は、ユーザが、1グループの値からこれに隣接のグループへと容易にブラウズすることができるようにする。これら望ましいフィーチャを備えていても尚、データを、どのテーブルの再生成も必要とせずにデータベースに加えることができる。さらに、サーチ  
20 することが非常に効率的であり、したがってサーバがユーザに関する状態情報を記憶することを必要とされない。

#### 発明の摘要

本発明にしたがい、ユーザは、サーチ値をサーバまたはサービス・プロバイダに提出する。このサーチ値は、探そうとしているデータに関しユーザが最も良いと思っているものである。ワイルドカードをユーザが入力することは必要でない。

サーバは、在来フォーマットでデータベースを保持する。このデータベースは、複数のフィールドを各々含む複数のレコードを有する。各レコードの少なくとも1つのフィールドには、キー・フィールドがあり、これは、データベースのそれらレコードをサーチするのに使用する。このデータベースに対し、指定のキー値を使用することにより、標準の質問  
30 を実行することができる。

また、サーバは、ページ定義テーブルに複数のページ定義も保持する。各ページ定義は、下側キー値と上側キー値でデータベース・レコードの1つのページを定める。1つの実施形態においては、各ページ定義は、単一の対の下側キー値および上側キー値を示す。サーチ値の受け取りに応答して、サーバは、それらページ定義を調べることにより、そのサーチ値を囲むキー値を有する1つを見つける。次に、これらキー値を使用してデータベースをサーチし、そしてそのページ定義のそれらキー値間に入るキー値を有する全てのレコード  
40 を見つける。

従来技術とは対照的に、ページ定義の上側キー値と下側キー値とは、ユーザに返すべきデータベース・レコードの最初と最後のキー値を指定しない。この代わりに、下側キー値は、返すべき最初のデータベース・レコードのキー値を指定する一方、上側キー値は後続のページのレコードの最初のデータベース・レコードのキー値を指定する。

別の実施形態においては、各ページ定義は、2つのキー値、すなわち、サーチ値に対応するページ定義を識別するのに使用する第1の対と、一旦正しいページ定義を見つけた後にデータベースに質問するのに使用する第2の対と、を有する。この実施形態を使用すると、ユーザに返す情報は、ページ定義を識別するのに使用するインデックス法に範囲が一致する必要がなくなる。これにより、クライアントに返す集合をより構成可能にできる。代表的には、第1対は、第2対が定める範囲の部分集合であるキー値範囲を定める。

#### 【図面の簡単な説明】

図1は、本発明による情報検索システムのブロック図。

図2は、本発明によるクライアントまたはサーバのコンピュータの関連するコンポーネン  
50

トを示すブロック図。

図3は、本発明による好ましいステップを示すフローチャート。

#### 好ましい実施形態の詳細な説明

図1は、本発明の1つの実施形態による情報検索コンピュータ・システムを示しており、全体を参照番号10で指示している。情報検索システム10は、サーバ12と、複数のクライアント14とを含んでいる。これらクライアントとサーバとの間の通信は、ローカル・エリア・ネットワーク、ワイド・エリア・ネットワークあるいはインターネットのようなパブリックのネットワークのようなデータ通信ネットワークを介して提供する。

サーバ12は、ハードディスクまたはハードディスク・アレイのような1つ以上のデータ記憶媒体を有するか、あるいはこれらへのアクセスを有する。サーバは、これらハードディスクの内の1つにデータベース16を保持している。このデータベースは、従来のフォーマットでフォーマットし配列して、複数のデータベース・レコードを有する。各レコードは、可変のデータを包含する複数のフィールドを有する。これらレコードに対し、1つ以上のキー・フィールド内に含まれた値に基づいて、サーチおよびソートの一方または両方を行うことができる。キー・フィールド内の値は、本文ではキー値と呼ぶ。サーバは、このデータベースを操作するためのデータベース管理システム(DBMS)18を有する。質問は、DBMSに対し、SQL(標準照会言語)フォーマットであるいはリレーショナル・データベースへのアクセスを可能にするそれと類似の言語で提出することができる。

図2は、全体を参照番号40で指示した例示のクライアント・コンピュータの関連するハードウェア・コンポーネントを示している。コンピュータ40は、セットトップ・ボックス、ハンドヘルド・コンピュータまたはポケット・コンピュータ、インターネット端末、デスクトップ・コンピュータ、およびラップトップ・コンピュータもしくはノートブック・コンピュータのような、コンシューマ機器を含む種々の異なったタイプのデバイスの内の任意の1つとすることができる。サーバ・コンピュータは、図2のコンフィギュレーションと同様のコンフィギュレーションとするが、ただし、ファイル・アクセス、アプリケーション・シェアリングおよびデータベース・アクセスのようなサーバ機能に対し最適化する。

コンピュータ40は、マイクロプロセッサ41と、電子メモリ42のようなコンピュータ読み取り可能な記憶媒体とを含む。プロセッサ40は、インテル社(Intel Corporation of Santa Clara, California)製造のx86シリーズのマイクロプロセッサのような従来の在庫がありすぐに入手可可能なマイクロプロセッサである。図2は、たった1つのプロセッサしか示していないが、本システムは、複数のプロセッサを含むこともでき、これらは、複数の異なったプロセスまたはタスクであってその各々が1つ以上の実行スレッドを有するものが使用する。また、コンピュータ・システム40は、他の代表的なコンピュータ・ハードウェア、例えばインターフェース電子回路を含むI/Oコンポーネント38、並びにオーディオおよびビデオのコンポーネントのようなマルチメディア・レンダリング・デバイス39も含む。

物理メモリ42は、好ましくは、ランダム・アクセス可能な読み書き電子メモリ(RAM)、並びに磁気ハードディスクあるいはその他のデータ記録媒体のような二次記憶装置から構成している。メモリ42はまた、他のタイプのコンピュータ読み取り可能な記憶媒体(これに実行可能なコンポーネントを分配させ記憶させる)も含むことができ、これは、フロッピー・ディスク、CD-ROM、EPROMのような不揮発性の電子メモリを含む。データベース16は、メモリ42内に、あるいはネットワーク・リソースを介してのみアクセス可能な他のディスク・ベースのメモリ内に格納することができる。

従来のコンピュータ・システムにしたがい、コンピュータ・システム40は、オペレーティング・システム43と、このオペレーティング・システムと共に実行する1つ以上のアプリケーション44とを備える。マイクロソフト社(Microsoft Corporation of Redmond, Washington)から入手可能なWindows 95 R オペレーティング・システムは、クライアント・コンピュータにおいて使用するのに適したオペレーティング・システムの1つの



例である。マイクロソフト社のWindows NT R オペレーティング・システムは、サーバ・コンピュータに適したオペレーティング・システムの1つの例である。

複数のアプリケーション・プログラムは、代表的には、メモリ42から実行する。クライアントにおいては、アクティブのアプリケーション・プログラムは、マイクロソフト社のInternet Explorer<sup>TM</sup>のようなインターネット・ブラウザを含む。サーバにおいては、アクティブのアプリケーションは、SQL<sup>®</sup> Server, OracleおよびInternet Information Server (インターネット情報サーバ)のようなデータベース管理システムを含む。

再び図1を参照すると、本発明は、サーバが保持するデータベース16内のデータを見つけ出す方法を包含している。一般には、クライアント14の1つは、サーチ値をサーバ12に提出する。このサーチ値は、データベース・レコードのキー・フィールドの内の1つに含まれた情報タイプのものである。サーバは、このサーチ値の提出に応答して、提出されたサーチ値に最も近く一致するであろうデータベース・レコードの限られた1つの集合を識別し、そしてそれからのデータを返す。このような1つのデータベース・レコード集合は、1ページのデータベース・レコードと呼び、このレコード・ページから返されたデータは、データ・ページと呼ぶ。

1つのサーチ値に応答してたった1つのレコード・ページを識別するため、サーバは、複数のページ定義を保持しそしてこれを参照する。各ページ定義は、少なくとも1対のキー値を示す。ある特定のサーチ値が与えられたときには、サーバは、ページ定義の内のそのサーチ値に対応する1つのものを識別するステップを実行する。次に、サーバは、データベースをサーチすることにより、識別したページ定義が示すキー値により限定 (bound) されたキー値を有する1つのページすなわち1つのデータベース・レコード集合を見つける。これらレコードからのデータは、次に、1つのデータ・ページとしてクライアントに返す。これらレコードは、オプションとして、ユーザに返す前にソートする。例えば、ネームは、ユーザに返す前に、アルファベット順にソートすることもできる。

ページ定義は、好ましくは、サーバによりアクセス可能でありかつDBMSがサービスする記憶媒体にサーチ可能なテーブルで格納し、そしてこれは、本文ではページ定義テーブルと呼ぶ。このようなテーブルは、図1に参照番号20で指示している。ページ定義テーブルは、複数の個別のエントリあるいは行を有する。各エントリは、1対のキー値、すなわち、下側 (LOWER) キー値および上側 (UPPER) キー値を含む。本発明のこの実施形態においては、これらキー値は、インデックス・キー値およびバウンド・キー値の両方として使用する。これらタイプのキー値間の差は、本発明の第2実施形態に関する以下の説明から明かとなる。

テーブル3は、上記説明において使用した例示のデータベースに対応するページ定義テーブルの1例を示している。

下側 (LOWER)	上側 (UPPER)
<b>a</b>	Beaver
Beaver	Bratsanos
Bratsanos	Cheng
Cheng	De Donato
De Donato	Dulfer
Dulfer	Fang
Fang	Francoeur
Francoeur	Gui
Gui	Hersey
Hersey	Kearney
Kearney	Lee
Lee	Lyons
Lyons	McMahon
McMahon	Moffit
Moffit	O'Conner
O'Conner	Prets
Prets	Roberts
Roberts	St. Clair
St. Clair	Troupe
Troupe	Watson
Watson	ZZZZZZZ

テーブル 3

このページ定義テーブルの各エントリは、このエントリの下側と上側のキー値により限定したキー値を有する 1 つのデータベース・レコード集合に対応している。より具体的には、ページ定義テーブルのある 1 つの特定のエントリに対応する単一のデータベース・レコード集合は、( a ) このエントリの下側キー値に等しいかあるいはこれより大きく、しかも ( b ) このエントリの上側キー値より小さい、キー値を有する全てのレコードから成っている。

上の例示のテーブル 3 において、第 1 のエントリは、“ a ” の下側キー値を有するが、それは、“ a ” が、このシステムにおける辞書学的に最初の可能なキー値であるからである（この値は、データのタイプに依存して異なる）。これは、“ Beaver ” の上側キー値を有する。したがって、このエントリに対応する 1 つのデータベース・レコード集合は、アルファベット順のリスト内の Beaver より前の全てのネームを含む。第 2 のエントリに対応する 1 つのデータベース・レコード集合は、“ Beaver ” と “ Bowman ” との間の範囲内の全てのネームを含む（“ Bowman ” は、データベース内の “ Bratsanos ” に先行するネームである）。一般的には、第 1 エントリに対する下側キー値は、任意の可能なサーチ値に等しいかあるいはこれより小さい値である。このテーブルの最後のエントリに対する上側キー値は、任意の可能なサーチ値に等しいかあるいはこれより大きい値である（再び、この値は、データベース内に含まれるデータのタイプに依存して異なる）。

ページ定義テーブルは、情報の所望のサイズおよび分割に基づいて定義する。アルファベット順の情報では、これは、アルファベットの各文字に対する単一のページを有することが望ましいことがあり、これは、ページ定義テーブルに対する必要を取り除くことになる。しかし、より粗いあるいはより微細の分割を有することにより、適当な量のデータあるいはおよそ等しい量のデータが各ページ・テーブル・エントリに対応するようにすることが望ましい場合がある。これは、大きなデータ集合に対しては特に当てはまる。単一のページ定義テーブルの良い例は、辞書または住所録のタブ ( A-B XYZ 等 ) である。

尚、このテーブルは、テーブル 1 内の上を示したテーブルとは異なっていることに注意されたい。例えばテーブル 3 は、ID 番号を指定しない。単に下側キー値と上側キー値とを指定する。加えて、各行の上側キー値は、次の行の下側キー値と同じである。これは、あ

10

20

30

40

50

る種のサーチ値がテーブル・エントリとテーブル・エントリとの間に入るのを防ぐ、すなわち、任意の可能なしかも許可されたサーチ値が、それらエントリの内の1つのキー値内に入ることになる。

クライアントがデータに対する要求の1部分としてサーチ値を提出したとき、サーバは、ページ定義テーブルをサーチすることにより、そのサーチ値に対応する1つの行あるいはエントリを識別する。詳細には、サーバは、そのサーチ値を囲む下側キー値と上側キー値とを有するエントリを見つける。さらに詳細には、サーバは、(a)そのサーチ値が当該エントリの下側キー値に等しいかあるいはそれより大きいこと、かつ(b)そのサーチ値が当該エントリの上側キー値よりも厳密に小さいこと、が当てはまる単一のエントリを見つける。

10

一旦、1つのエントリを見つけると、そのキー値をバウンド値として使用することにより、その対応するデータベース・レコード集合を識別する。詳細には、サーバは、データベースのサーチまたは質問を実行して、その識別したテーブル・エントリのキー値により限定したキー値を有するデータベース・レコード集合を識別する。この集合は、(a)当該エントリの下側キー値に等しいかあるいはこれより大きく、かつ(b)当該エントリの上側キー値よりも小さい、キー値を有する全てのレコードから成っている。これらレコードからのデータは、クライアントのデータ要求への応答として返す。

ページ定義テーブルはキー値をリストするが、データベースは、ほとんどの場合、キー・フィールド以外の複数のフィールドを有することになる。クライアントに返す値は、必ずしも、その識別したデータベース・レコード集合からのキー値である必要はない。むしろ、一旦適当なデータベース・レコード集合を識別したならば、任意のグループのフィールド値を、データベース・レコード集合から返すことができる。しかし、ほとんどの場合、各レコードに対し返すそのグループのフィールド値は、キー・フィールドからの値を含むことになる。

20

所望のデータを返すことに加え、サーバは、そのデータベース・レコード集合を識別するのに使用した下側キー値および上側キー値を返す。この状態情報は、次に、クライアントが記憶し、そしてクライアントが使用する最初に見つけた集合に隣接する追加の集合またはページのデータベース・レコードを見つけるための後続の質問または要求を定式化するのにクライアントが使用する。これは、クライアントが保持する状態情報に基づいてデータベース情報の諸ページ間をブラウズできるようにし、この状態情報を保持するためにサーバ・リソースを消費することはない。加えて、ある範囲内に入る多くのレコードがある場合(例えば、ベートーベン(Beethoven)から始まる作品は、音楽データベースでは1000のエントリを含み、したがって100の集合が返される)、この技術により、本システムは、良好に機能し、しかも依然としてユーザがそのデータをブラウズできるようにする。

30

尚、好ましくはサーチ値の何等かの事前処理があることに注意されたい。例えば、全ての値は、小文字に変換し、しかも先行するスペースは除去する。“The”または“A”のようなキー・ワードもまた除去する。ここで、オリジナルのデータは、ある種のフィールドに対する“キー”値を有することがあることに注意されたい。例えば、オリジナル・データが“The Beatles”であるとした場合、これは、サーチのために使用する追加の列“beatles, the”も含むことができる。この例においては、そのサーチ・キー・データは、返さないが、オリジナルの“実際”のデータを返すことになる。ページ定義テーブル内の値は、これと同じ方法で処理する。

40

図3は、本発明による好ましいステップを示す。ステップ100は、サーバからクライアントへのサーチ値の提出から成っている。このサーチ値は、サーバが保持するデータベース内の1つのキー・フィールドに対応している。このサーチ値の提出に回答して、サーバは、複数のページ定義を参照することによりその供給されたサーチ値に対応するページ定義の内の1つを識別するステップ102を実行する。既に説明したように、各ページ定義は、1対のキー値を示し、これにおいて、各対は、上側キー値と下側キー値とを含んでいる。ページ定義は、好ましくは1つのテーブル内にエントリとして格納する。これが当て

50

はまるときには、ステップ 102 は、供給されたサーチ値を囲む 1 対のキー値を有するテーブル・エントリをサーチすることから成っており、ここで、そのサーチ値は、下側キー値と上側キー値とにより定まる下側キー値を含む範囲内に入る。

ステップ 104 は、データベースをサーチすることにより、ステップ 102 で識別したページ定義のキー値により限定されたキー値を有する 1 つのデータベース・レコード集合を見つけることから成る。詳細には、このステップは、その識別したページ定義の下側キー値と上側キー値との間（下側キー値を含む）のキー値を有する全てのレコードを返すように定式化する。

ステップ 106 は、識別したデータベース・レコード集合からのデータをクライアントに返すことから成る。このステップは、ステップ 102 で識別したページ定義からのキー値を返すことを含んでいる。クライアントは、この返されたキー値に基づいて新たなサーチ値を得ることができる。このような新たなサーチ値は、データベース・レコードの異なったページ中をブラウズするために、サーバに対し提出する。

本技法は、上述の従来技術の技法に対する大きな改善となるものである。その最も顕著な利点は、レコードをテーブルに追加するときに、テーブルの再生成を全く必要としないことである。レコードをデータベースに追加することは、事実上、異なった数のレコードを有するレコード・ページをもたらすことがあるが、これは、重要な関心事ではない。大きなデータベースでは、相対的なサイズは、およそ等しいままであり、小さな統計学上の変動を受けるに過ぎない。あるポイントでは、全てのレコード・ページをおよそ等しいサイズにセットするため、ページ定義テーブルを再生成あるいは再計算することが望ましくなることがある。しかし、これは、比較的小さなタスクであり、従来技術において必要とされていたページ・データ・テーブルを再生成するタスクに比較すれば特にである。加えて、ページ定義テーブルの定義または再生成のときに、データベース・レコードのランダムなサンプリングを使用することにより、等しいサイズのレコード集合を定義することができる。これは、さらに、再生成に必要な時間を減少させる。もっと重要なことは、ページ・データ・テーブルが必要でないことである。データベース自体への参照は、下側と上側のバウンド（bound）をページ定義テーブルから識別した後に、この技術を使って行う。ここで、サーチは、多数のページ定義テーブルを保持することにより、異なったデータベース・フィールドに基づかせることができる。これは、従来技術では多数のページ・データ・テーブルを必要とするものであった。

この技法を使用するサーチは、素早くかつ効率的であって、ページ定義テーブル内の簡単なインデックス付きルックアップと、データベース内の後続のインデックス付き質問のみを必要とするだけである。サーチ値として“fle”を使用する例示の質問は、SQL フォーマットでは以下のように現れる。

```
select LOWER, UPPER from page_definition_table
```

```
where LOWER <= 'FLE' and UPPER > 'FLE'
```

```
select name, address from database_table where
```

```
key_value >= 'FANG' and key_value < 'FRANCEOUR'
```

ここで、key\_value は各レコードに対するキー値を含むデータベース・フィールドを指し、database\_table は、この既存のデータベース内の 1 つの適当なテーブルを指す。“境界”条件に関する適当なチェックは、行を全く返さない質問をチェックすることにより行う。

ここで、2 つの先行する質問文の内の後者は、データベース・レコードに含まれる任意のデータを、どのようなテーブルの再生成も必要とせずに、返すことができることに注意されたい。その返すフィールドを予め定義することは、従来技術におけるのと同じように必要はない。この場合、その質問は、選択したページのデータベース・レコード内における各レコードに対しフルネームと住所とを返すように定式化する。

これら質問は、サーバにより定式化しそして提出するか、あるいはさもなければ、これら

10

20

30

40

50

は、クライアントから直接定式化しそして提出することができる。好ましい実施形態においては、質問は、サーバが実行し、そしてデータをHTMLフォーマットで返す。

本発明の別の重要な特徴は、サーバではなくクライアントが保持する状態情報を使用して、1つのデータ・ページからこれに隣接のデータ・ページへブラウズすることができるようにするである。この理由は、データベースの質問を定式化するのに使用するキー値をクライアントに与えるからである。例えば、上記例では、“Fang” “Francoer” が返されたことになる。前の集合またはページのデータへとブラウズするには、クライアントは、これら値の内の1つに基づいて新たなサーチを提出することになる。例えば、前のページのデータベース・レコードに対する新たな上側キー値と下側キー値とは、以下の形態の質問を実行することにより見つける。

```
select LOWER, UPPER from page_definition_table
```

```
where UPPER = previous_lower
```

後続の1つのデータベース・レコードに対する上側キー値および下側キー値は、以下の通りに見つける。

```
select LOWER, UPPER from page_definition_table
```

```
where LOWER = previous_upper
```

本発明が対処する別の問題は、返されたデータについてのユーザに対し都合の良い提示に関するものである。クライアントがある特定のレコードをサーチしているとき、このレコードは、一般に、返されかつソートされたデータ内にランダムに配置される。それは、その返されたレコードの初めに、あるいはその返されたレコードの終わりに、もしくはその中間にあることがある。しかし、所望ならば、ページ定義テーブルは、より小さなエントリに分割することにより、各データ・ページが比較的小さなものとなるようにすることができる。このとき、複数のデータ・ページをクライアントに返すことができ、そのデータ・ページが、返されたページ内に任意の配置されたサーチ値に対応する。この方法を使用すると、サーチ値に最も近い一致を、返されたデータの初め付近、真ん中付近、あるいは終わり付近に一貫して配置することができるが、ただし、代表的には、初め付近にそのデータを表示することが望ましいであろう。例えば、一旦ターゲットのデータ・ページを決定したなら、このデータ・ページおよびこれに続くデータ・ページを返すことができる。さらにユーザの便宜のため、ユーザ指定のキー値に最も近い一致を強調することも望ましい。

以上に説明した技法に対する強化法として、ページ定義が、2対のキー値、すなわち、1対のインデックス・キー値と、1対のパウンド・キー値とを含むように構成することができる。インデックス・キー値は、サーチ値に対応するページ定義を見つけるのに使用し、そしてパウンド・キー値は、クライアントに返すべきデータベース・レコードのパウンドを定めるのに使用する。このようにして、返されるレコードの範囲は、こうしなければ上述の技法を使って返されることになる範囲を超えて拡張することができる。この代替技法は、初め、真ん中、終わりの付近のような、返されるデータ内の任意の場所付近に、最も近く一致するレコードを配置するために使用することもできる。

以下のテーブル4は、本発明のこの代替の実施形態によるページ定義テーブルを示している。

10

20

30

40

インデックス 下側 (INDEX LOWER)	インデックス 上側 (INDEX UPPER)	バウンド 下側 (BOUNDING LOWER)	バウンド 上側 (BOUNDING UPPER)
a	Beaver	a	Beckett
Beaver	Bratsanos	Barnes	Browns
Bratsanos	Cheng	Borgman	Conway
Cheng	De Donato	Caudill	DeBroeck
De Donato	Dulfer	Davis	Dunkelberger
Dulfer	Fang	Driscoll	Ferguson
Fang	Francoeur	Evans	Funaro
Francoeur	Gui	Flowers	Gwertzman
Gui	Hersey	Gower	Honeycutt
Hersey	Kearney	Heflin	Kelnhofer
Kearney	Lee	Jones	Lim
Lee	Lyons	Lalonde	Maes
Lyons	McMahon	Ludden	McCrory
McMahon	Moffit	Maumas	Moncure
Moffit	O'Conner	Milford	Oker
O'Conner	Prets	Naroski	Raines
Prets	Roberts	Planondon	Rodriquez
Roberts	St. Clair	Richardson	Sturm
St. Clair	Troupe	Slonsky	Tutt
Troupe	Watson	Timoney	Webb
Watson	ZZZZZZZ	Walker	ZZZZZZZ

テーブル 5

ページ定義テーブルの各エントリは、1 対のインデックス・キー値と 1 対のバウンド・キー値との両方を含んでいる。インデックス・キー値は、あるサーチ値にどのテーブル・エントリが対応するかを識別するときに使用する。詳細には、下側のインデックス・キー値がサーチ値に等しいかそれ未満でありかつその上側インデックス・キー値がそのサーチ値よりも大きいエントリは、このサーチ値に対応するエントリである。

これらインデックス・キー値は、テーブル 3 内に示したのと全く同じように構成する。すなわち、それら対が順番になったシーケンスを形成し、これにおいて、ある特定の対の上側キー値は、後続の対の下側キー値と同じである。これは、サーチ値がインデックス・キー値の各対間に入るのを防止する。

バウンド・キー値は、1 つのレコード集合を求めてデータベースに質問するときに使用する。一旦、テーブル・エントリを識別したなら、そのデータベースに対し、その識別したテーブル・エントリのバウンド・エントリ間に入るキー値を有する全てのレコードを求めて質問する。

この技法を使用するとき、ある特定のテーブル・エントリのバウンド値は、好ましくは同じエントリのインデックス・エントリに渡る。例えば、テーブル 4 の第 3 エントリについて考えてみる。そのインデックス・キーは、“Bratsanos” と “Cheng” とである。これら値 (“Bratsanos” を含む) 間のサーチ値は、その結果として、このテーブル・エントリが選択されることになる。しかし、返されるデータは、“Borgman” と “Conway” との間のキー値を有する全てのデータベース・レコードに対応することになる。したがって、バウンド範囲は、どの特定のエントリに対するインデックス範囲よりも大きい。ある種の場合には、ある特定のエントリに対するバウンド・キーは、このエントリに対するインデックス・キー値と同じとすることもできる。この特別な場合は、テーブル 3 の前の例により実際に実施しており、これにおいては、インデックス・キー値とバウンド・キー値とは、互いに等しく、一方の対の値のみをページ定義テーブル内にリストすることしか必要でな

10

20

30

40

50

い。

この方法を使用すると、サーチ値に最も近く一致するデータは、そのサーチ結果の1つの部分に向かって偏らせることができる。例えば、下側バウンド・キー値が下側インデックス・キー値に等しく、かつ上側バウンド・キー値が上側インデックス・キー値よりもはるかに大きい場合、最も近い一致は、返されるデータの初めに向かって偏らせることになる。

。

データを返すとき、バウンド・キーおよびインデックス・キーのいずれか、あるいはその両方をクライアントに返す。これにより、クライアントは、上述のように、別の異なったデータベース・レコード集合にブラウズするため、それら値に基づいて、新たなサーチ規

10

準をサーバに提出することができる。好ましくは、返されたインデックス・キー値は、新たなサーチ質問を定式化するのに使用する。

この強化法は、クライアントが指定したキー値に対し相対的に、クライアントに返されるデータ集合を制御するのに有用である。この他にも利点がある。先ず、サーバは、クライアントに返されたデータ内におけるサーチしたワードの位置を制御することができる。もし、キー値（またはその最も近い一致）が返される最初のレコードとなるように希望する場合、それより前のレコードは、単に、返すサーチ結果から省くことができる。言い換えれば、それら前のレコードは、第2の質問の結果から、これをクライアントに返す前に“切り捨てる”ことができる。これを行うためには、“前の”レコードを選択することが、切り捨てたそれらレコードを返さなければならない。こうすることにより、前の1つのレコード集合に対する第1の要求が、おそらくその前レコードを返すことにならず、むしろ

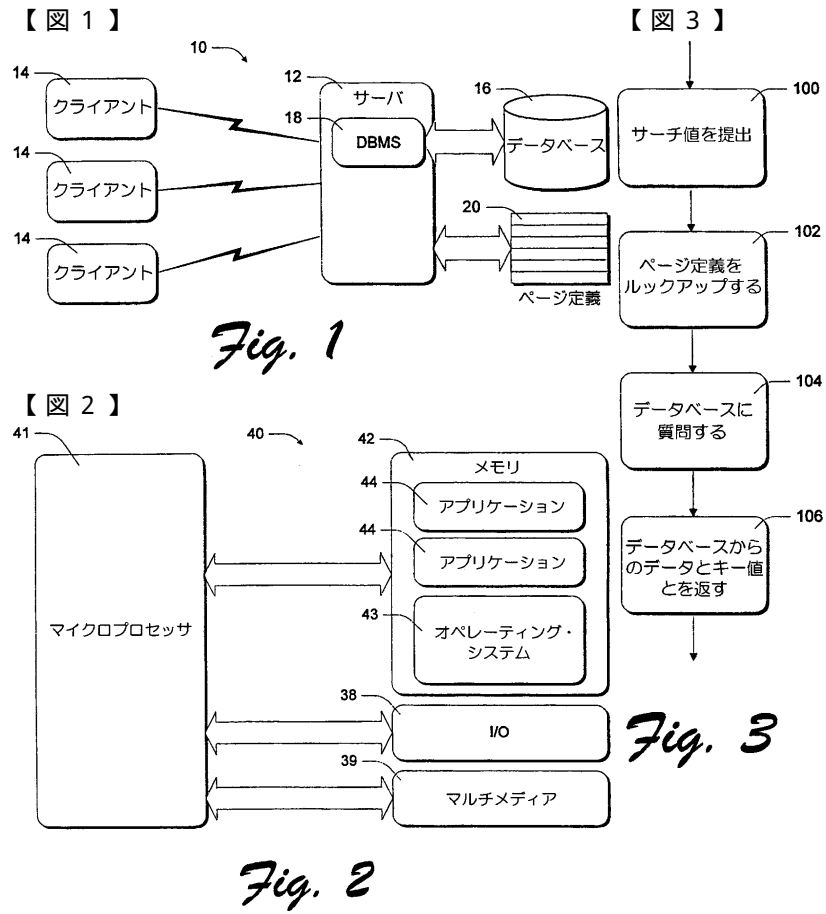
20

オリジナルの切り捨てられていない最初の集合を返すことになる。これは、クライアントが記憶しそしてサーバに送り戻す単一の状態フラグで実現することができる。第2に、ブラウズすることは、データ集合をオーバーラップさせることにより強化することができる。これにより次または前の集合をビューする間においてあるコンテキストが存在するようにする。

法令に従い、本発明について多少なりとも具体的な用語でその構造および方法上の特徴について以上に説明した。しかし、理解されるべきであるが、本発明は、記述したこの特定の特徴に限定されるものではないが、それは、本文に開示した手段は、本発明を実施に移す好ましい形態を構成するものであるからである。したがって、本発明について、均等論に従い適切に解釈される添付の請求の範囲の適正な範囲内に入るその形態および変更のい

30

ずれにおいても、その保護を請求するものである。





---

フロントページの続き

- (72)発明者 ブリン, アーノルド・エヌ  
アメリカ合衆国ワシントン州 9 8 0 0 4 , クライド・ヒル, ノース・イースト・トゥエンティセヴ  
ンス・ストリート 9 4 0 1
- (72)発明者 ロートン, マイケル・エス  
アメリカ合衆国カリフォルニア州 9 4 1 1 0 , サンフランシスコ, ウィンドフィールド・ストリー  
ト 3 1

審査官 鈴木 和樹

- (56)参考文献 特開平 0 3 - 0 0 4 3 6 6 ( J P , A )  
特開平 0 5 - 3 1 4 1 8 8 ( J P , A )  
特開昭 6 4 - 0 4 8 1 2 9 ( J P , A )  
特開平 0 5 - 3 3 4 1 6 5 ( J P , A )  
特開平 0 7 - 1 9 1 8 8 6 ( J P , A )  
Directional Key Support for Closed End Boundary Checking, IBM Technical Disclosure Bul  
letin, 1 9 9 4 年 7 月, 第 3 7 巻, 第 7 号, p . 4 8 9 - 4 9 1

- (58)調査した分野(Int.Cl. , D B 名)  
G06F 17/30  
G06F 12/00  
NRIサイバーパテント