

(19) 日本国特許庁(JP)

(12) 特 許 公 報(B2)

(11) 特許番号

特許第4101410号
(P4101410)

(45) 発行日 平成20年6月18日(2008.6.18)

(24) 登録日 平成20年3月28日(2008.3.28)

(51) Int.Cl.

F I

G O 6 F 12/00 (2006.01)

G O 6 F 12/00 5 1 7

G O 6 F 17/30 (2006.01)

G O 6 F 12/00 5 0 5

G O 6 F 17/30 2 4 0 A

請求項の数 4 (全 12 頁)

(21) 出願番号	特願平11-237979	(73) 特許権者	590003191
(22) 出願日	平成11年8月25日(1999.8.25)		フジツウ サービス リミテッド
(65) 公開番号	特開2000-76105(P2000-76105A)		イギリス国、ダブリュ1ユー 3 ビーダブ
(43) 公開日	平成12年3月14日(2000.3.14)		リュ ロンドン、ベカー ストリート 2
審査請求日	平成16年11月12日(2004.11.12)		2
(31) 優先権主張番号	9818819.6	(74) 代理人	100064447
(32) 優先日	平成10年8月29日(1998.8.29)		弁理士 岡部 正夫
(33) 優先権主張国	英国 (GB)	(74) 代理人	100085176
			弁理士 加藤 伸晃
		(74) 代理人	100106703
			弁理士 産形 和央
		(74) 代理人	100096943
			弁理士 臼井 伸一
		(74) 代理人	100091889
			弁理士 藤野 育男

最終頁に続く

(54) 【発明の名称】 タイムバージョンデータ記憶装置

(57) 【特許請求の範囲】

【請求項 1】

データ処理装置であって、

(a) 複数のレコードを記憶する手段であって、該レコードの各々が、

(i) 少なくとも1つの属性、

(i i) 前記属性が有効な時間の範囲を示すタイムスパン、

(i i i) 前記レコードがいつ作成されたかを示す挿入時間、及び

(i v) 前記レコードがコンクリートレコードであるのか、あるいは、直前のレコードの属性値を修正する可能性のあるデルタレコードであるのかを示すタイプフィールドを含む、

記憶する手段、並びに、

(b) 既存の属性値及び既存のタイムスパンを有する既存レコードを新たな属性値及び新たなタイムスパンに更新する手段であって、該更新が、前記新たなタイムスパンを前記既存のタイムスパンと比較し、該比較の結果に応じて、多数の異なる所定の動作のうちの1つを実行することからなり、前記所定の動作が、

(i) 前記新たなタイムスパンが前記既存のタイムスパンよりも前にある場合には、前記新たなタイムスパンにデルタレコードを付加すること、

(i i) 前記新たなタイムスパンが前記既存のタイムスパンの終了時を包含する場合には、前記新たなタイムスパンにコンクリートレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの開始時に終わるように修正すること、

(i i i) 前記新たなタイムスパンが前記既存のタイムスパンに完全に包含される場合には、前記新たなタイムスパンに第 1 のコンクリートレコードを付加し、前記新たなタイムスパンの終了時に続く該既存のタイムスパンの部分に第 2 のコンクリートレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの開始時に終わらせること、

(i v) 前記新たなタイムスパンが前記既存のタイムスパンを完全に包含する場合には、該既存のタイムスパンの開始時に続く新たなタイムスパンの部分にコンクリートレコードを付加し、該既存のタイムスパンに先行する該新たなタイムスパンの部分にデルタレコードを付加し、該既存レコードのタイムスパンを削除すること、及び

(v) 前記新たなタイムスパンが前記既存のタイムスパンの開始時を包含する場合には、該新たなタイムスパンの終了時に続く該既存のタイムスパンの部分にコンクリートレコードを付加し、該新たなタイムスパンにデルタレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの終了時に終わるように修正することからなる、更新する手段

からなるデータ処理装置。

【請求項 2】

挿入時間が視点時間よりも遅くないレコードのみを検索し、検索されたレコードから属性値を構成することにより、指定された視点時間の視点から属性値を検出する手段を含む請求項 1 記載のデータ処理装置。

【請求項 3】

データ処理方法であって、

(a) データベースが複数のレコードを記憶するステップであって、該レコードの各々が、

(i) 少なくとも 1 つの属性、

(i i) 前記属性が有効な時間の範囲を示すタイムスパン、

(i i i) 前記レコードがいつ作成されたかを示す挿入時間、及び

(i v) 前記レコードがコンクリートレコードであるのか、あるいは、直前のレコードの属性値を修正する可能性のあるデルタレコードであるのかを示すタイプフィールドを含む、

記憶するステップ、並びに、

(b) 更新機能部が、既存の属性値及び既存のタイムスパンを有する既存レコードを新たな属性値及び新たなタイムスパンに更新するステップであって、該更新するステップが、前記新たなタイムスパンを前記既存のタイムスパンと比較するステップ、及び該比較の結果に応じて、多数の異なる所定の動作のうちの 1 つを実行するステップからなり、前記所定の動作が、

(i) 前記新たなタイムスパンが前記既存のタイムスパンよりも前に場合には、前記新たなタイムスパンにデルタレコードを付加すること、

(i i) 前記新たなタイムスパンが前記既存のタイムスパンの終了時を包含する場合には、前記新たなタイムスパンにコンクリートレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの開始時に終わるように修正すること、

(i i i) 前記新たなタイムスパンが前記既存のタイムスパンに完全に包含される場合には、前記新たなタイムスパンに第 1 のコンクリートレコードを付加し、前記新たなタイムスパンの終了時に続く該既存のタイムスパンの部分に第 2 のコンクリートレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの開始時に終わらせること、

(i v) 前記新たなタイムスパンが前記既存のタイムスパンを完全に包含する場合には、該既存のタイムスパンの開始時に続く新たなタイムスパンの部分にコンクリートレコードを付加し、該既存のタイムスパンに先行する該新たなタイムスパンの部分にデルタレコードを付加し、該既存レコードのタイムスパンを削除すること、及び

(v) 前記新たなタイムスパンが前記既存のタイムスパンの開始時を包含する場

10

20

30

40

50

合には、該新たなタイムスパンの終了時に続く該既存のタイムスパンの部分にコンクリートレコードを付加し、該新たなタイムスパンにデルタレコードを付加し、該既存レコードのタイムスパンを該新たなタイムスパンの終了時に終わるように修正することからなる、更新するステップ
からなるデータ処理方法。

【請求項 4】

挿入時間が視点時間よりも遅くないレコードのみを検索し、検索されたレコードから属性値を構成することにより、指定された視点時間の視点から属性値を検出するステップを含む請求項 3 に記載のデータ処理方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】

本発明は、タイムバージョンデータ記憶装置に関するものである。

【0002】

【従来の技術】

(例えば、関係データベースなどの)従来のデータ記憶装置では、データ記憶装置への問合せが行なわれると、最後の更新日に、データの現行状態を反映した結果が返却される。また、既存バージョンシステムは、変更は必ず現行状態に対して行なわれるものであり、過去や将来の変更はサポートされていないという前提のうえに成り立っている。

【0003】

【発明が解決しようとする課題】

しかしながら、将来のある選択された地点からデータの状態を見ることができの方が好ましい。さらに、現行状態だけでなく、過去または将来のシステム状態を変更できることが好ましい。ところが、従来のデータ記憶装置は、そのような機能を有していない。

【0004】

そこで本発明は、将来の任意の選択された地点からデータの状態を見る機能をサポートする新規データ記憶装置を提供することを目的とする。

【0005】

【課題を解決するための手段】

本発明によれば、複数のレコードを記憶する手段からなるデータ処理装置において、各レコードが、

(a)少なくとも1の属性と、

(b)前記属性が有効なタイムスパンを示すタイムスパンと、

(c)レコードがいつ作成されたかを示す挿入時間と、

(d)そのレコードが、コンクリートレコードか、あるいは、直前のコンクリートレコードまたはデルタレコードの属性値を修正する可能性のあるデルタレコードであるかを示すタイプフィールドとを具備するデータ処理装置。

【0006】

【発明の実施の形態】

本発明によるタイムバージョンデータ記憶装置を具備する1データ処理装置について、添付図面を参照し、例を示しながら以下に説明する。

【0007】

このデータ記憶装置では、オブジェクトのタイムバージョンビューをサポートしている。その中心にあるのは、将来のある特定の地点からオブジェクト状態を見ることができ的能力であり、「1960年に行なわれていれば、どのような評価結果が得られたか」などの問合せをサポートする能力である。そこには、次の2つの主要な概念がある。

1)視点時間 T_V 。これは、状態が観測されている時間的な地点である。

2)目標タイムスパン $[T_S, T_E]$ 。これは、データが必要とされるタイムスパンである。

【0008】

例えば、 $[T_S, T_E]$ と T_V が、いずれも「1960年1月1日」であれば、ちょうど1

10

20

30

40

50

960年現在のシステム状態を表していることになる。 T_V が1995年1月1日であり、 $[T_S, T_E]$ が共に1960年1月1日であれば、1960年現在のシステム状態であるものの、1995年までに1960年の状態に対して(例えば、1960年の状態に見られるエラーを修正するなどの)何らかの変更が加えられることを示している。また、さらに、例えば、「1999年1月1日から、XはYに居住している予定である」という事実を通知された場合など、オブジェクトの将来の状態について検討するうえで適している。

【0009】

前述したような行為をサポートするためには、オブジェクト状態への修正はすべて、修正が行なわれた時間と修正が適用される時間に必ず対応づけられていなければならない、また、どの情報も必ず修正過程で失われてはならない。

10

【0010】

さらに実際に重要になる点は、タイムバージョン状態(オンラインとアーカイブのいずれも)を維持するうえで必要な記憶量と、記憶データからオブジェクト状態に関する特定のビューを構成する際に伴う処理コストのいずれに換算した場合も、アプローチに関するコストである。以下の説明は、次の各事項を仮定して行なわれている。

【0011】

1)オブジェクト状態の通常の使用パターンは、過去または将来の状態ではなく、「現行状態」である。現行状態は、通常、「現在からXまで」ではなく、「今後」の状態である。

【0012】

20

2)通常の更新パターンは、過去または将来の状態への変更ではなく、現行状態への変更である。ここで、変更は、通常、「現在からXまで」ではなく、「今後」適用される。

【0013】

3)一定の時期よりも古いデータに対しては、アーカイビング法を使用しなければならない。

【0014】

[システム概要]

図1について説明すると、本装置は、コンピュータシステム10からなり、データベース11と、データベースのデータにアクセスするクライアントアプリケーション12を具備している。クライアントは、更新機能13と読取り機能14により、データベースにアクセスする。さらに、本装置は、アーカイブ記憶装置15を具備している。

30

【0015】

[データベース]

データベースは、多数のレコード(または行)からなり、各レコードは、以下のフィールド(または列)から構成されている。

【0016】

UniqueID: この一意名によって、レコードに対応するオブジェクトが識別される。例えば、顧客や従業員などの特定の人を識別するのに用いられる。

【0017】

Attributes: レコードに対応するオブジェクトの1つ以上の属性。例えば、レコードに、人の名前、給料、誕生日等を表す属性が含まれていてもよい。さらに、「なし」(値がないことが明らかな場合を示している)や「不明」(値があるか否か明らかなことを示している)を表す特殊な値が含まれていてもよい。

40

【0018】

StartTime: このレコードに含まれているデータが定義されるタイムスパンの開始時点。

【0019】

EndTime: このレコードに含まれているデータが定義されるタイムスパンの終了時点。

【0020】

50

`InsertTime` : このレコードが作成された日付け。

【0021】

`Type` : レコード型。この値には、「`concrete`」、「`delta`」、「`archive`」の3つの値のうちのいずれか一つを使用できる。

【0022】

本例では、`StartTime`、`EndTime`、`InsertTime`が、単純データ形式(`dd/mm/yyyy`)で定義されている。特殊なデータ値である「`top`」と「`bottom`」は、仮定された最大および最小限度の日付けを示すためのものである。

【0023】

「`concrete`」レコードは、指定されたタイムスパンにわたるオブジェクトの属性値を示すレコードである。コンクリートレコードの`StartTime`値は、非包含的な値である。すなわち、その値は、タイムスパンが実際に開始する1日前に相当する。これに対し、`EndTime`フィールドは、包含的な値であり、タイムスパンの実際の終了時に等しい。その効果は、隣接する実際のレコードがある場合、あるレコードの`EndTime`が次のレコードの`StartTime`に等しくなることにある。

【0024】

「`delta`」レコードは、通常、既存のコンクリートレコードや可能性として考えられる他のデルタレコードに関して、(適正でない)属性値のサブセットを捕獲するレコードである。また、その影響を受けない属性のフィールドエントリに、「`unused`」という値がある。「`concrete`」レコードと異なり、「`delta`」レコードの`StartTime`は、包含的な値である。

【0025】

「`archive`」レコードは、指定されたアーカイブポイント時間 T_A を含むそれまでの挿入時間を持つ値がアーカイブされたことを示している。これらの値にアクセスする必要があるときは、アーカイブ記憶装置に問い合わせなければならない。

【0026】

以下の説明と図では、明快さを考慮して、`UniqueID`フィールドが省略されており、各オブジェクトには単一の属性値しか含まれていないものと仮定している。

【0027】

[データベースの更新]

更新機能に関する以下の説明において、引用されているすべてのコンクリートレコードの開始時間は、実際の開始時間であり、(実際の開始時間よりも1少ない)記録された開始時間ではない。このことから、論理アルゴリズムが影響を受けないことがさらに理解しやすくなる。

【0028】

図2に、データベース更新機能について示されている。ここでは、目標タイムスパン $[T_S, T_E]$ の新規の値 Y で特定の属性を更新しなければならない状態が仮定されている。

【0029】

(ステップ21) 更新機能は、まず初めに、「`current time`」に T_V を設定し、設定タイムスパン $[T_S, T_E]$ にわたって、下記のデータベース読取り機能(ステップ41、42、44、および45のみ)を使用する。これによって、0またはそれ以上のコンクリートレコードが返却される。興味深い値として、最新のコンクリートレコード C_1 がある。この値は、属性値 X とタイムスパン $[T_{START}, T_{END}]$ を有していると仮定されている。このような既存のコンクリートレコードが検出されない場合は、その更新が属性を作成する前の時点に適用されていることが暗に示されている。

【0030】

(ステップ22) 既存のコンクリートレコードが検出されない場合や、このような既存レコードのタイムスパンが「トップ」まで達していない場合は、属性値 Y とタイムスパン $[T_S, T_E]$ を有する単一のデルタレコードを作成して、更新が行なわれる。

10

20

30

40

50

【 0 0 3 1 】

そうでない場合は、状況はさらに複雑なものとなり、以下に説明する通り、目標タイムスパン $[T_S, T_E]$ とクリティカルポイント T_{START} との関係によって異なっている。

【 0 0 3 2 】

図 3 A、3 B、3 C、3 D、3 E 及び 3 F を参照しながら、以下の説明を行なう。これらの図では、コンクリートレコードの開始時間として記憶される値は、実際の開始時間を 1 下回る値である。

【 0 0 3 3 】

図 3 A は、値 X とタイムスパン $[06 / 06 / 1996, top]$ を有する既存のコンクリートレコードを示している。(ここで、前述した通り、コンクリートレコードの $StartTime$ は、非包含的な値であり、各コンクリートレコードのタイムスパンの実際の開始は、コンクリートレコードの $StartTime$ 値よりも実際には 1 日後である。) したがって、この例において、クリティカルポイント T_{START} は、 $06 / 06 / 1996$ である。図 3 B 乃至 3 F では、各種目標タイムスパン $[T_S, T_E]$ に対して、この既存レコードを新規の値で更新した結果を示している。

【 0 0 3 4 】

(ステップ 23) 目標タイムスパン $[T_S, T_E]$ が、クリティカルポイント T_{START} よりも完全に前にある場合、属性値 Y とタイムスパン $[T_S, T_E]$ を有する単一のデルタレコードを作成することによって、更新が行なわれる。例えば、図 3 B は、目標タイムスパンが $[04 / 04 / 1994, 05 / 05 / 1995]$ の場合を示している。

【 0 0 3 5 】

(ステップ 24) 目標タイムスパン $[T_S, T_E]$ が、クリティカルポイント T_{START} よりも後にあり、「トップ」まで達している場合、属性値 Y とタイムスパン $[T_S, top]$ を有する単一のコンクリートレコードを作成することによって、更新が行なわれる。既存コンクリートレコードの $EndTime$ 値は、 $T_S - 1$ に変更される。例えば、図 3 C は、目標タイムスパンが $[07 / 07 / 1997, top]$ の場合を示している。

【 0 0 3 6 】

(ステップ 25) 目標タイムスパン $[T_S, T_E]$ が、クリティカルポイント T_{START} よりも後にあるが、「トップ」まで達していない場合、属性値 Y とタイムスパン $[T_S, T_E]$ を有する第 1 のコンクリートレコードと、属性値 X とタイムスパン $[T_E + 1, top]$ を有する第 2 のコンクリートレコードを作成することによって、更新が行なわれる。既存コンクリートレコードの $EndTime$ 値は、 $T_S - 1$ に変更される。図 3 D は、目標タイムスパンが $[07 / 07 / 1997, 08 / 08 / 1998]$ の場合を示している。

【 0 0 3 7 】

(ステップ 26) 目標タイムスパン $[T_S, T_E]$ が、クリティカルポイント T_{START} よりも後にあり、「トップ」まで達している場合、属性値 Y とタイムスパン $[T_{START} + 1, top]$ を有するコンクリートレコードと、属性値 Y とタイムスパン $[T_S, T_{START}]$ を有するデルタレコードを作成することによって、更新が行なわれる。既存コンクリートレコードの $EndTime$ 値は、 T_{START} に変更される。例えば、図 3 E は、目標タイムスパンが $[05 / 05 / 1995, top]$ の場合を示している。

【 0 0 3 8 】

(ステップ 27) 上記以外の場合で、目標タイムスパン $[T_S, T_E]$ がクリティカルポイント T_{START} に及んでいるものの、「トップ」まで達していない場合は、属性値 X とタイムスパン $[T_E + 1, top]$ を有するコンクリートレコードと、属性値 Y とタイムスパン $[T_S, T_E]$ を有するデルタレコードを作成することによって、更新が行なわれる。既存コンクリートレコードの $EndTime$ 値は、 T_E に変更される。例えば、図 3 F は、目標タイムスパンが $[05 / 05 / 1995, 07 / 07 / 1997]$ の場合を示している。

【 0 0 3 9 】

10

20

30

40

50

ただし、既存レコードが更新されるいずれの場合も、更新が、当該レコードの $EndTime$ 値の変更に限定されている。このような変更は、コンクリートレコードに対してのみ発生し、その場合、必ず「top」からそれを下回る値 T_x への変更である。さらにまた、新規コンクリートレコードは、必ずタイムスパン $[T_x + 1, top]$ にわたって作成される。

【0040】

また、すでに説明した通り、レコードの更新、すなわち、適正なフィールドに特殊な「none」値を挿入することによって値を削除できる。

【0041】

このような更新プロセスを用いれば、データベースへの更新中に情報が失われることはないということが明らかである。新規データは、新規データが旧データと入れ代わっていることが推測できるような方法で記録される。

10

【0042】

[作成]

目標タイムスパン $[T_s, T_e]$ に対して属性を値 Y で作成することは、上記の更新機能で説明した $X[T_{start}, T_{end}]$ (すなわち、 C_i) の代わりに、「unknown (不明)」 $[bottom - 1, top]$ な仮定上のコンクリートレコードを更新することに等しい。その結果、ステップ24またはステップ25のどちらかを実行することになり、1または2個のコンクリートレコードを作成することになる。ただし、仮定レコードの T_{end} への変更は、無視される。

20

【0043】

[データベースへのアクセス]

図4ではデータベース読取り機能が示されている。この機能により、時間 T_v の視点から、指定されたタイムスパン $[T_s, T_e]$ の属性値を知ることができる。

【0044】

(ステップ41) 読取り機能によって、データベースへのアクセスが行なわれ、以下の条件を満足するレコードをすべて検索する。

$InsertTime \leq T_v$ であり、

$EndTime \geq T_s$ であり、

$StartTime \leq T_e$ である。

30

【0045】

この検索によって、0またはそれ以上のコンクリートレコードまたはアーカイブレコードと、0またはそれ以上のデルタレコードが返却される。検索されたレコードは、次のように処理される。(ステップ42乃至47について以下に説明する処理は、検索されたレコードのコピー上で実行され、元の記憶レコードは変更されないことは理解できるであろう。)

【0046】

(ステップ42) アーカイブレコードが検出された場合、アーカイブ記憶装置15に問合せが行なわれ、そのレコードは、アーカイブ記憶装置からの対応するレコードで置き換えられる。このプロセスは、アンパッキングを必要とするアーカイブレコードがなくなるまで、再帰的に繰り返される。

40

【0047】

(ステップ43) 最終コンクリートレコードの終了時間 T_e が「top」でない場合は、「top」に設定される。(このレコードの終了時間を「top」以外のものに変更する更新であれば、その更新は、 T_v よりも後に行なわれたはずであり、したがって、その場合を考慮する必要はない。)

【0048】

(ステップ44) 最終コンクリートレコードの開始時間が T_e であれば、そのレコードは廃棄される。

【0049】

50

(ステップ45) 残存しているすべてのコンクリートレコードの開始時間が1だけ増分され、包含的な開始時間が示される。

【0050】

(ステップ46) 検索されたデルタレコードは、挿入時間の降順に分類される。

【0051】

(ステップ47) 次にレコードは、時間順に時間範囲を走査し、時間の各ポイントに対して、最終InsertTimeを有するレコードを取り出し、その値を読み取って処理される。

【0052】

このようなプロセスは、基本的な連続するコンクリートレコードの最上部にデルタレコードが積み重ねられ、次に、上から下へ値を見ながら各値を読み取って、最初に出会った値を取り出す形にヴィジュアル化することが可能である。(ただし、同一期間中に、デルタレコードの挿入時間に等しいかそれよりも遅い挿入時間を有するコンクリートレコードは一切存在しない。)コンクリートレコードとデルタレコードのどちらも使用されない期間があれば、この期間中の値は、「unknown」となる。

【0053】

例えば、図5では、一連のコンクリートレコードC1、C2とデルタレコードD1、D2、D3が示されている。各タイムスパンは、水平軸に沿って表されており、挿入時間は、垂直軸に沿って表されている。この例では、読取り機能によって、次のような一連の値が返却される。

C1 [t1, t2 - 1]

D1 [t2, t3 - 1]

D2 [t3, t5 - 1]

D3 [t5, t7]

C2 [t7 + 1, t8]

【0054】

[アーカイピング]

アーカイピングにより、オブジェクト状態の旧部分をニアラインまたはオフライン記憶装置に格納することにより、主オブジェクト記憶装置の必要記憶量を最小限度に抑えられる。

【0055】

アーカイピングは、定期的に行なわれるアーカイピングプロセスによって実行される。アーカイピングプロセスでは、データベースをすべて走査し、指定されたアーカイブポイント時間 T_A に等しいかそれよりも早期の挿入時間と終了時間を持つレコードを検出する。これらの各レコードは、アーカイブ記憶装置15に書き込まれ、データベース11内の単一のアーカイブレコードで置き換えられる。

【0056】

このアーカイブレコードには、次のようなフィールドがある。

InsertTime = "bottom"

StartTime = "bottom"

EndTime = T_A

属性 = アーカイブ記憶装置内のアーカイブ済みレコードの位置を示すアーカイブ参照。

【0057】

このプロセスは、アーカイブレコード自体をアーカイブしながら、繰り返し使用できる。

【0058】

ただし、アーカイブ期間に使用されるもののアーカイブポイント後に書き込まれたデルタレコードが、アーカイピングプロセスによって取り除かれることはない。

【0059】

(幾つかの可能性ある変更態様)

10

20

30

40

50

前述のシステムに多くの変更を行なっても、本発明の範囲を逸脱しないことは、明らかである。

【 0 0 6 0 】

例えば、データの読取り時に、最も最近の値から、時間を増分させながら値を返却できる。この方法の最大の利点は、アーカイブデータのアンパッキングを、絶対的に必要な状態になるまで延期できる点にある。データのクライアントが、 $[T_S, T_E]$ のサブセットだけが必要であると判断した場合、この利点によって大幅な節約が可能になる。

【 0 0 6 1 】

記録された開始時間が実際の開始時間よりも 1 だけ下回る場合、コンクリートレコードを使用する方法を変更して、実際の開始時間を使用することができる。この変更によって生じる影響は、レコードの読取り時に、時間の余分な 1 単位を考慮しなければならないことであり、余分なコンクリートレコードだけでなく不要なデルタレコードも検出される可能性がある。(このような事態は、問合せ論理を用いることによって回避できるが、すべての読込みにおいて、そのためのコストが増大することになる。)

【 0 0 6 2 】

コンクリートレコードをすべて取り除き、デルタレコードだけを使用することも可能である。ただし、この方法は、システムが必要とする記憶量を削減する一方で、オブジェクト状態と複雑なアーカイピングのアセンブルにかかる処理コストが増大する。

【図面の簡単な説明】

【図 1】本発明を具体的に示したデータ処理装置のブロック図である。

【図 2】データベース更新機能を示すフローチャートである。

【図 3 A】データベース更新機能の各動作例を示す図である。

【図 3 B】データベース更新機能の各動作例を示す図である。

【図 3 C】データベース更新機能の各動作例を示す図である。

【図 3 D】データベース更新機能の各動作例を示す図である。

【図 3 E】データベース更新機能の各動作例を示す図である。

【図 3 F】データベース更新機能の各動作例を示す図である。

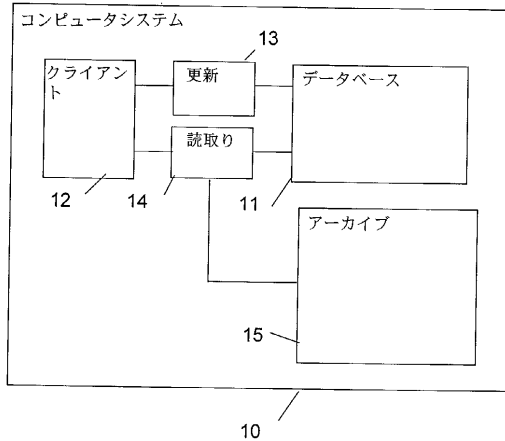
【図 4】データベース読取り機能を示すフローチャートである。

【図 5】データベース読取り機能の動作を示す図である。

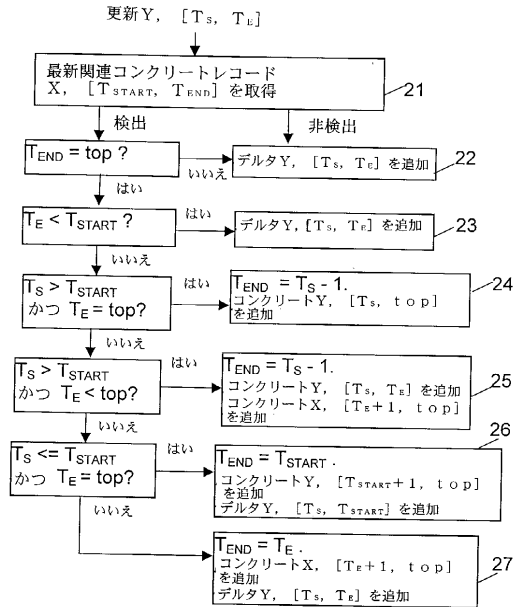
【符号の説明】

- 1 0 コンピュータシステム
- 1 1 データベース
- 1 2 クライアント
- 1 3 更新
- 1 4 読取り
- 1 5 アーカイブ

【図 1】



【図 2】



【図 3 A】

既存データ・クリティカルポイント = 06/06/1996

属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	トップ	01/01/1997	コンクリート
Y	04/04/1994	05/05/1995	01/01/1998	デルタ

【図 3 B】

更新 Y, [04/04/1994, 05/05/1995] - クリティカルポイントより前

属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	トップ	01/01/1997	コンクリート
Y	04/04/1994	05/05/1995	01/01/1998	デルタ

【図 3 C】

更新 Y, [07/07/1997, top] - クリティカルポイントより後、トップまで

属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	06/07/1997	01/01/1997	コンクリート
Y	06/07/1997	トップ	01/01/1998	コンクリート

【図 3 D】

更新 Y, [07/07/1997, 08/08/1998] - クリティカルポイントより後、トップまで達していない

属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	06/07/1997	01/01/1997	コンクリート
Y	06/07/1997	08/08/1998	01/01/1998	コンクリート
X	08/08/1998	トップ	01/01/1998	コンクリート

【図 3 E】

更新 Y, [05/05/1995, top] - クリティカルポイントにかかる、トップまで

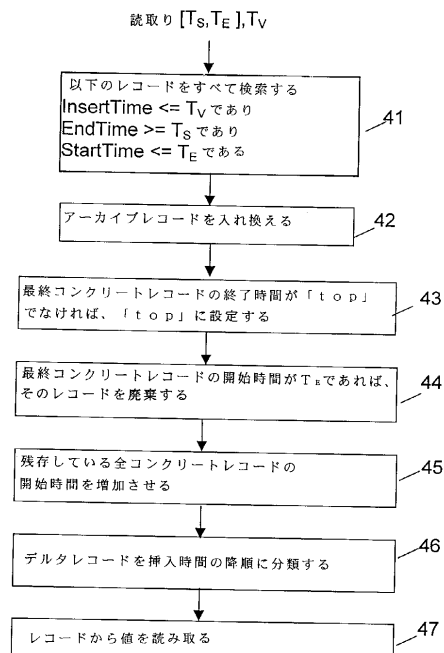
属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	06/06/1996	01/01/1997	コンクリート
Y	06/06/1996	トップ	01/01/1998	コンクリート
Y	05/05/1995	06/06/1996	01/01/1998	デルタ

【図 3 F】

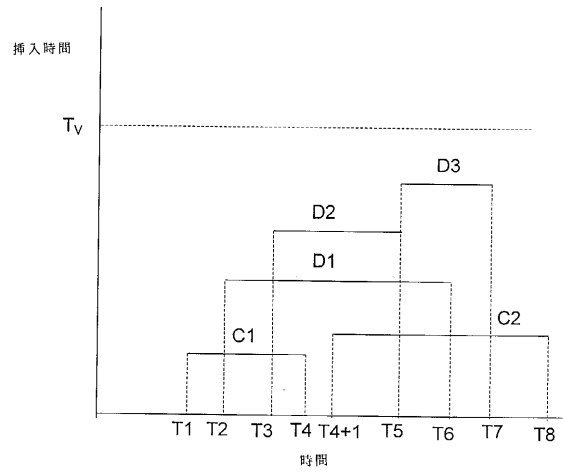
更新 Y, [05/05/1995, 07/07/1997] - クリティカルポイントにかかる、トップまで達していない

属性	開始時間	終了時間	挿入時間	レコード型
X	05/06/1996	07/07/1997	01/01/1997	コンクリート
X	07/07/1997	トップ	01/01/1998	コンクリート
Y	05/05/1995	07/07/1997	01/01/1998	デルタ

【図 4】



【図 5】



フロントページの続き

- (74)代理人 100101498
弁理士 越智 隆夫
- (74)代理人 100096688
弁理士 本宮 照久
- (74)代理人 100102808
弁理士 高梨 憲通
- (74)代理人 100104352
弁理士 朝日 伸光
- (74)代理人 100107401
弁理士 高橋 誠一郎
- (74)代理人 100106183
弁理士 吉澤 弘司
- (74)代理人 100101432
弁理士 花村 太
- (74)代理人 100106161
弁理士 竹島 智司
- (72)発明者 ボール アントン リチャードソン ガードナー
イギリス国 . エスエル 1 7 エッチテー バッキンガムシャー バーンハム セント ピーターズ
クローズ 5 2

審査官 高瀬 勤

- (56)参考文献 特開平 0 9 - 2 6 5 4 1 9 (J P , A)
特開平 0 3 - 2 8 0 1 3 8 (J P , A)

- (58)調査した分野(Int.Cl. , D B 名)
G06F 12/00
G06F 17/30
JSTPlus(JDream2)