(19)日本国特許庁(JP)　(12)公開特許公報(A)　(11)特許出願公開番号
特開2012-155498
(P2012-155498A)
(43)公開日　平成24年8月16日(2012.8.16)

(51)Int.Cl.		F I			テーマコード（参考）
G06F 12/00	(2006.01)	G06F	12/00	535G	5B082
		G06F	12/00	533F	

審査請求 未請求　請求項の数 7　ＯＬ　（全 24 頁）

(21)出願番号　特願2011-13476(P2011-13476)
(22)出願日　平成23年1月25日(2011.1.25)

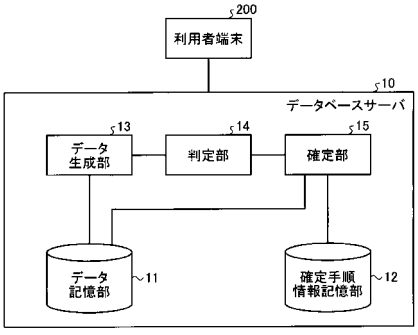
(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号
(74)代理人　100089118
弁理士　酒井　宏明
(72)発明者　武部　信幸
神奈川県川崎市中原区上小田中４丁目１番１号　富士通株式会社内

Ｆターム(参考)　5B082 FA16　GB01　GB06

(54)【発明の名称】データベースサーバ装置、データベース更新方法及びデータベース更新プログラム

(57)【要約】
【課題】競合するトランザクションを同時に実行する。
【解決手段】データベースサーバは、記憶されるデータを更新する複数のトランザクションの間で各トランザクションを確定させる確定手順情報を記憶する。また、データベースサーバは、データを更新するトランザクションを受付けた場合に、トランザクションを実行して、データから確定前の更新データを生成する。また、データベースサーバは、トランザクションを確定させる場合に、データを更新元データとする確定済みデータを生成した確定済みトランザクションが存在するか否かを判定する。そして、データベースサーバは、確定済みトランザクションが存在する場合に、確定済みデータと確定手順情報とに従って確定前の更新データを再び更新して、トランザクションを確定させる。
【選択図】図１

実施例1にかかるデータベースサーバの構成を示すブロック図



【特許請求の範囲】
【請求項１】
データを記憶するデータ記憶部と、
前記データ記憶部に記憶されるデータを更新する複数のトランザクションの間で各トランザクションを確定させる確定手順情報を記憶する確定手順情報記憶部と、
前記データを更新するトランザクションを受付けた場合に、当該トランザクションを実行して、前記データから確定前の更新データを生成するデータ生成部と、
前記データ生成部によって実行されたトランザクションを確定させる場合に、前記データを更新元データとする確定済みデータを生成した確定済みトランザクションが存在するか否かを判定する判定部と、
前記判定部によって確定済みトランザクションが存在すると判定された場合に、前記確定済みデータと前記確定手順情報とに従って前記確定前の更新データを再び更新して、前記トランザクションを確定させる確定部と、
を有することを特徴とするデータベースサーバ装置。
【請求項２】
前記データ記憶部は、複数のデータ項目を有するレコードを前記データとして記憶し、
前記確定手順情報記憶部は、前記データ項目ごとに前記確定手順情報を記憶し、
前記データ生成部は、前記レコードを更新するトランザクションを受付けた場合に、当該トランザクションを実行して、前記データ項目を更新した確定前の更新レコードを生成し、
前記判定部は、前記データ生成部によって実行されたトランザクションを確定させる場合に、前記レコードを更新元のレコードとする確定済みレコードを生成した確定済みトランザクションが存在するか否かを判定し、
前記確定部は、前記判定部によって確定済みトランザクションが存在すると判定された場合に、前記確定済みレコードと前記確定手順情報とに従って、前記確定前の更新レコードの各データ項目を再び更新して、前記トランザクションを確定させることを特徴とする請求項１に記載のデータベースサーバ装置。
【請求項３】
前記データ記憶部が記憶するレコードは、当該レコードを更新元レコードとする前記確定済みレコードを特定する特定情報を有し、
前記判定部は、前記データ生成部によって実行されたトランザクションを確定させる場合に、前記レコードが有する特定情報に基づいて、前記確定済みレコードを生成した確定済みトランザクションが存在するか否かを判定することを特徴とする請求項２に記載のデータベースサーバ装置。
【請求項４】
前記データ記憶部が記憶するレコードは、前記データ項目ごとに、当該データ項目が更新された否かを示す更新フラグを有し、
前記確定部は、前記判定部によって確定済みトランザクションが存在すると判定された場合に、前記更新フラグに基づいて更新されたデータ項目を特定し、前記確定済みレコードと前記確定手順情報とに従って、特定したデータ項目を再び更新して、前記トランザクションを確定させることを特徴とする請求項２に記載のデータベースサーバ装置。
【請求項５】
前記確定部は、前記トランザクションを確定させる場合に、当該トランザクションを確定させる前記確定済みデータに対する他のトランザクションを排他し、当該トランザクションを確定した場合に、前記確定済みデータに対する他のトランザクションの排他を解除することを特徴とする請求項２に記載のデータベースサーバ装置。
【請求項６】
データベースサーバ装置が
データを更新するトランザクションを受付けた場合に、当該トランザクションを実行して、前記データから確定前の更新データを生成するデータ生成ステップと、

前記データ生成ステップによって実行されたトランザクションを確定させる場合に、前記データを更新元データとする確定済みデータを生成した確定済みトランザクションが存在するか否かを判定する判定ステップと、
前記判定ステップによって確定済みトランザクションが存在すると判定された場合に、前記確定済みデータと複数のトランザクションの間で各トランザクションを確定させる確定手順情報とに従って前記確定前の更新データを再び更新して、前記トランザクションを確定させる確定ステップと、
を実行することを特徴とするデータベース更新方法。
【請求項７】
コンピュータに、
データを更新するトランザクションを受付けた場合に、当該トランザクションを実行して、前記データから確定前の更新データを生成するデータ生成手順と、
前記データ生成手順によって実行されたトランザクションを確定させる場合に、前記データを更新元データとする確定済みデータを生成した確定済みトランザクションが存在するか否かを判定する判定手順と、
前記判定手順によって確定済みトランザクションが存在すると判定された場合に、前記確定済みデータと複数のトランザクションの間で各トランザクションを確定させる確定手順情報とに従って前記確定前の更新データを再び更新して、前記トランザクションを確定させる確定手順と、
を実行させることを特徴とするデータベース更新プログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
本発明は、データベースサーバ装置、データベース更新方法及びデータベース更新プログラムに関する。
【背景技術】
【０００２】
従来、データベースサーバは、トランザクションを利用者端末から受付ける。データベースサーバは、トランザクションを受付けると、データの格納、データの読み出し、データの更新及び削除などの各種処理を実行する。このようなデータベースサーバは、同一のデータを更新するトランザクションを複数の利用者から同時に受付けた場合には、データの整合性を保証するためにロック等の手法で各種処理を排他する。
【０００３】
すなわち、データベースサーバは、同一のデータを更新するトランザクション、言い換えると、競合するトランザクションを排他的に制御することでデータの整合性を保証する。例えば、データベースサーバは、悲観的ロックや楽観的ロックなどの排他制御を用いてトランザクションを制御する。
【０００４】
悲観的ロックは、「データベース内のあるデータを更新するトランザクションを同時に複数の利用者から受付ける可能性が高い」ことを前提にした方法であり、トランザクションが更新対象のデータを読み出した時点で他の更新トランザクションを排他する。
【０００５】
また、楽観的ロックは、「データベース内のあるデータを更新するトランザクションを同時に複数の利用者から受付ける可能性が少ない」ことを前提にした方法であり、トランザクションが更新対象のデータを読み出してから更新するまでの間に、他のトランザクションがそのデータを更新してコミットしていた場合に、当該トランザクションをエラーにする。
【０００６】
このように、悲観的ロックや楽観的ロックなどの排他制御を利用したデータベースサーバは、他のトランザクションを排他することで、競合するトランザクションを逐次化して

データの整合性を保証している。
【０００７】
また、データベースサーバが複数のトランザクションをまとめて処理する技術も開示されている。例えば、データベースサーバは、ある一定期間に入力されたトランザクションをまとめて、それぞれのトランザクションで読み出し及び更新するデータ毎に再合成することで実行順序を決定し、トランザクションを実行する。
【先行技術文献】
【特許文献】
【０００８】
【特許文献１】特開２００９－２７１６６５号公報
【発明の概要】
【発明が解決しようとする課題】
【０００９】
しかしながら、上述した従来の技術では、競合するトランザクションを同時に実行することができないという課題があった。例えば、排他制御を用いたデータベースサーバでは、競合するトランザクションを逐次化することでデータの整合性を保証するので、競合するトランザクションを同時に実行することができない。
【００１０】
また、複数のトランザクションをまとめて処理する方法では、データベースサーバは、例えば、分岐する命令を判定するトランザクションを実行する場合、トランザクション内でデータを検索した結果によって条件を判定して次の処理を実行するので、トランザクション内で実行する処理の全てをデータベースに事前に送付することができない。アプリケーションの設計段階で、同時に実行中の他のトランザクションからの更新を考慮して、各トランザクションを実行するように設計することは困難であり、現実的ではない。
【００１１】
本発明は、競合するトランザクションを同時に実行することができるデータベースサーバ装置、データベース更新方法及びデータベース更新プログラムを提供することを目的とする。
【課題を解決するための手段】
【００１２】
データを更新する複数のトランザクションの間で、更新が競合した場合に、各トランザクションを確定させる確定手順情報を記憶する手段を設ける。データベースサーバは、データを更新するトランザクションを受付けた場合に、トランザクションを実行して、更新元データから確定前の更新データを生成する。そして、データベースサーバは、トランザクションを確定させる場合に、更新元データから確定済みデータを生成した他の確定済みトランザクションが存在するか否かを判定する。そして、データベースサーバは、他の確定済みトランザクションが存在する場合に、更新元データと確定済みデータと確定手順情報とに従って確定前の更新データを再び更新して、トランザクションを確定させる。
【発明の効果】
【００１３】
競合するトランザクションを同時に実行することができる。
【図面の簡単な説明】
【００１４】
【図１】図１は、実施例１にかかるデータベースサーバの構成を示すブロック図である。
【図２】図２は、実施例２にかかるデータベースサーバの構成を示すブロック図である。
【図３】図３は、確定手順情報ＤＢが記憶する情報の一例を示す図である。
【図４】図４は、確定手順情報ＤＢバッファが記憶する情報の一例を示す図である。
【図５】図５は、データベースがテーブル「TBL1」として記憶する情報の一例を示す図である。
【図６】図６は、データベースバッファが記憶するトランザクションのレコードセットの

一例を示す図である。
【図７Ａ】図７Ａは、データ操作部がアクセス手順として生成したＳＱＬ文の一例を示す図である。
【図７Ｂ】図７Ｂは、データ操作部がアクセス手順として生成したＳＱＬ文の別例を示す図である。
【図７Ｃ】図７Ｃは、データ操作部がアクセス手順として生成したＳＱＬ文の別例を示す図である。
【図８】図８は、更新確定部によるデータの確定を示す図である。
【図９】図９は、更新確定部がトランザクションをコミットした後のデータベースの状態の一例を示す図である。
【図１０】図１０は、更新確定部による処理の処理手順を示すフローチャートである。
【図１１】図１１は、マージ処理の処理手順を示すフローチャートである。
【図１２】図１２は、レコード操作部によって読み取り可能と判定されたレコードＩＤをトランザクション毎に示す図である。
【図１３】図１３は、データベース更新プログラムを実行するコンピュータシステムを示す図である。
【発明を実施するための形態】
【００１５】
以下に、本願の開示するデータベースサーバ装置、データベース更新方法及びデータベース更新プログラムの実施例を図面に基づいて詳細に説明する。なお、この実施例によりこの発明が限定されるものではない。
【実施例１】
【００１６】
図１は、実施例１にかかるデータベースサーバ１０の構成を示すブロック図である。図１に示すように、データベースサーバ１０は、データ記憶部１１と確定手順情報記憶部１２とデータ生成部１３と判定部１４と確定部１５とを有する。このデータベースサーバ１０は、利用者端末２００とネットワークを介して接続されており、データの検索、更新、削除などの各種処理を実行するトランザクションを受付ける。
【００１７】
データ記憶部１１は、データを記憶する。確定手順情報記憶部１２は、データ記憶部１１に記憶されるデータを更新する複数のトランザクションの間で各トランザクションを確定させる確定手順情報を記憶する。
【００１８】
データ生成部１３は、データを更新するトランザクションを受付けた場合に、当該トランザクションを実行して、データから確定前の更新データを生成する。判定部１４は、データ生成部１３によって実行されたトランザクションを確定させる場合に、データを更新元データとする確定済みデータを生成した確定済みトランザクションが存在するか否かを判定する。確定部１５は、判定部１４によって確定済みトランザクションが存在すると判定された場合に、確定済みデータと確定手順情報とに従って確定前の更新データを再び更新して、トランザクションを確定させる。
【００１９】
データ生成部１３は、トランザクション毎に新たなデータを更新し、同一のデータから別の新たなデータを生成したトランザクションが存在する場合に、確定部１５は、確定手順情報に従って、トランザクションと他トランザクションを確定させる。したがって、実施例１にかかるデータベースサーバ１０は、競合するトランザクションを同時に実行し、データの整合性を高めることができる。
【実施例２】
【００２０】
［実施例２にかかるデータベースサーバの構成］
次に、実施例２にかかるデータベースサーバの構成を説明する。図２は、実施例２にか

かるデータベースサーバ１００の構成を示すブロック図である。図２に示すように、実施例２にかかるデータベースサーバ１００は、通信制御Ｉ／Ｆ部１１０と確定手順情報ＤＢ１２１と確定手順情報ＤＢバッファ１２２とデータベース１２３とデータベースバッファ１２４とログＤＢ１２５と制御部１３０とを有する。このデータベースサーバ１００は、利用者端末２００とネットワークを介して接続されており、データの検索、更新、削除などのトランザクションを受付ける。なお、確定手順情報ＤＢ１２１と確定手順情報ＤＢバッファ１２２とデータベース１２３とデータベースバッファ１２４とログＤＢ１２５とは、例えば、半導体メモリ素子、又はハードディスクなどの記憶装置である。

【００２１】

通信制御Ｉ／Ｆ部１１０は、少なくとも１つの通信ポートを有するインターフェースであり、他の装置と間でやり取りされる情報を制御する。例えば、通信制御Ｉ／Ｆ部１１０は、データベース１２３に記憶されるデータを参照するトランザクションを利用者端末２００から受付け、データベース１２３の情報を出力する。また、通信制御Ｉ／Ｆ部１１０は、図示しない管理装置からの要求に応じて、確定手順情報ＤＢ１２１やログＤＢ１２５等に記憶される各種情報を管理装置に出力する。

【００２２】

確定手順情報ＤＢ１２１は、データベースのデータを更新する複数のトランザクションの間で各トランザクションを確定させる確定手順情報をデータベースのテーブルのデータ項目ごとに記憶する。図３は、確定手順情報ＤＢ１２１が記憶する情報の一例を示す図である。例えば、図３に示すように、確定手順情報ＤＢ１２１は、「列名、列のデータ型、デフォルト値、列の制約情報、更新競合時の属性」として「カラム1、Integer、0、NOT NULL、Conflicting Operation Max」を記憶する。同様に、確定手順情報ＤＢ１２１は、「カラム2、Char(1)、-、-、Conflicting Operation OverWrite」、「カラム3、Char(5)、-、-、Conflicting Operation OverWrite」、「カラム4、Integer、-、-、Conflicting Operation Merge」を記憶する。

【００２３】

ここで、「列名」は、データ項目の列名を示し、例えば、「カラム1」、「カラム2」、「カラム3」、「カラム4」などが格納される。「列のデータ型」は、格納される値の形式を指定する情報を示す。例えば、「列のデータ型」には、整数を定義する「Integer」や指定した文字数以下の文字列を定義する「Char」などが格納される。

【００２４】

「デフォルト値」は、後述する列の制約情報として「NOT NULL」が定義されている場合に、列のデータ型に対応して設定される初期値の値を示す。例えば、列のデータ型が「Integer」の場合には、「デフォルト値」は、「0」が格納される。「列の制約情報」は、データベースに無効なデータが入力されないように定義された制約についての情報を示す。例えば、「列の制約情報」には、NULL値の入力を禁止する「NOT NULL」が格納される。

【００２５】

「更新競合時の属性」は、トランザクションが競合した場合に、各トランザクションを確定させる確定手順情報を示す。例えば、「更新競合時の属性」には、更新結果の差分を計算する「Merge」、コミット時刻が後の結果で上書きする「OverWrite」、より大きい値を設定する「Max」、より小さい値を設定する「Min」などが格納される。また、「更新競合時の属性」には、他トランザクションによって更新されていれば、トランザクションをエラーに設定する「Error」、利用者が定義した関数を呼び出して利用者論理でマージする「Function」などが格納される。なお、更新競合時の属性についての詳細は後述する。

【００２６】

図３の例では、確定手順情報ＤＢ１２１は、「カラム1」には、小数点を持たない精度の整数値が格納され、「Not NULL」制約によって初期値は「0」に設定されることを示す。また、「カラム1」のデータ更新が競合した場合には、より大きい値が設定されることを示す。同様に、「カラム2」には、１文字以下の文字列が格納され、データ更新が競合

した場合には、コミット時刻が後の結果で上書きされることを示す。「カラム3」には、５文字以下の文字列が格納され、データ更新が競合した場合には、コミット時刻が後の結果で上書きされることを示す。「カラム4」には、小数点を持たない精度の整数値が格納され、データ更新が競合した場合には、差分を計算した値が設定されることを示す。

【００２７】

確定手順情報ＤＢバッファ１２２は、確定手順情報ＤＢ１２１が記憶する情報の一部もしくは全てを一時的に記憶する。図４は、確定手順情報ＤＢバッファ１２２が記憶する情報の一例を示す図である。例えば、図４に示すように、確定手順情報ＤＢバッファ１２２は、「列数」と列数毎に「列名」と「列のデータ型」と「デフォルト値」と「列の制約情報」と「更新競合時の属性」とを記憶する。ここで、「列数」は、カラムの数を示す。なお、「列名」と「列のデータ型」と「デフォルト値」と「列の制約情報」と「更新競合時の属性」とは、確定手順情報ＤＢ１２１が記憶する情報と同様であるので詳細な説明を省略する。

【００２８】

データベース１２３は、各種情報を複数のデータ項目を有するレコードとして記憶する。図５は、データベース１２３がテーブル「TBL1」として記憶する情報の一例を示す図である。例えば、図５に示すように、データベース１２３は、「レコードID、生成者ID、更新者ID、コミット時刻、世代ポインタ、更新フラグ1、カラム1、更新フラグ2、カラム2、更新フラグ3、カラム3、更新フラグ4、カラム4」を対応付けて記憶する。

【００２９】

ここで、「レコードID」は、レコードを一意に識別する識別子であり、例えば「R2」が格納される。「生成者ID」は、レコードを生成した生成者やトランザクションを一意に識別する識別子であり、例えば、「10」が格納される。「更新者ID」には、レコードを追加、レコードを更新、レコードを削除するトランザクションのいずれかが実行された場合、レコードを追加、レコードを更新、レコードを削除するトランザクションを識別する識別子が格納される。したがって、「更新者ID」が入っているレコードは、そのIDが示すトランザクションで更新後のレコードに置き換えられたり、削除されたりして無効になったことを意味する。なお、レコードを削除するトランザクションが実行された場合、「更新者ID」には、「生成者ID」と同じトランザクションを識別する識別子の値が格納される。「コミット時刻」は、トランザクション終了時刻を示す時刻情報であり、例えば、「10:00」が格納される。「世代ポインタ」には、レコードを更新して新たなレコードを生成したレコードを識別するレコードＩＤが格納される。

【００３０】

「カラム１、カラム２、カラム３、カラム４」は、確定手順情報ＤＢ１２１に記憶される確定手順情報の列のデータ型に従って設定されたデータ値を示し、例えば、「100、A、Init、5000」が格納される。

【００３１】

「更新フラグ1」は、カラム１に格納される情報が更新されて生成されたものであることを示す。例えば、「更新フラグ1」には、カラム１に格納される値が更新された場合に「y」が格納される。また、「更新フラグ1」には、カラム１に格納される値が更新されていない場合に「n」が格納される。「ｎ」が格納されていると言うことは、そのレコードが更新された時に当該のカラムは更新されず更新元の値を引き継いだことを意味する。なお、「更新フラグ2」、「更新フラグ3」及び「更新フラグ4」については「更新フラグ1」と同様であるので、詳細な説明は省略する。

【００３２】

図５の例では、データベース１２３は、「レコードID=R2」で識別されるレコードは、生成者ID「10」で識別されるトランザクションによって「10:00」に作成されたデータ値が「100、A、Init、5000」であり、データ値の全てが新たに作成されてコミットされたことを示す。同様に、データベース１２３は、「レコードID=R4」で識別されるレコードは、生成者ID「10」で識別されるトランザクションによって「10:00」に作成されたデータ

値が「30、F、Init、3200」であり、データ値の全てが新たに作成されてコミットされたことを示す。

【００３３】

また、データベース１２３は、「レコードID=R6」で識別されるレコードは、生成者ID「40」で識別されるトランザクションによってカラム２とカラム４の値が更新されたデータ値が「100、B、Init、4000」であることを示す。また、データベース１２３は、「レコードID=R7」で識別されるレコードは、生成者ID「50」で識別されるトランザクションによってカラム１～カラム４の値が更新されたデータ値が「20、C、Init、450」であることを示す。

【００３４】

また、データベース１２３は、「レコードID=R8」で識別されるレコードは、生成者ID「50」で識別されるトランザクションによってカラム３とカラム４の値が更新されたデータ値が「100、A、Upd、4000」であることを示す。また、データベース１２３は、「レコードID=R9」で識別されるレコードは、生成者IDと更新者IDとがともに「50」であるので、「50」で識別されるトランザクションによって作成され、何れかのレコードが削除されたことを示す。

【００３５】

データベースバッファ１２４は、データベース１２３に記憶される情報の更新対象であるレコードのコピーやレコードのコピーから生成した更新レコードを一時的に記憶する。なお、データベースバッファ１２４が記憶する情報は、図５に一例を示した、データベース１２３が記憶する情報と同じであるので、説明を省略する。

【００３６】

また、データベースバッファ１２４は、更新や削除する元になったレコードと、更新や削除追加により生成したレコードとのアドレスをレコードセットとして記憶する。図６は、データベースバッファが記憶するトランザクションのレコードセットの一例を示す図である。例えば、図６に示すように、データベースバッファ１２４は、「トランザクションID、更新元レコードID、生成レコードID、処理」を対応付けて記憶する。

【００３７】

ここで、「トランザクションID」は、トランザクションを識別する識別子であり、例えば「40」が格納される。「更新元レコードID」は、更新する元や削除する元になったレコードを一意に識別する識別子であり、例えば「R2」が格納される。「生成レコードID」は、更新、削除及び追加によって生成したレコードを一意に識別する識別子であり、例えば「R6」が格納される。「処理」は、トランザクションの種類がレコードを追加するトランザクションであるか、レコードを更新するトランザクションであるか、レコードを削除するトランザクションであるかを示し、例えば「更新」が格納される。

【００３８】

図６に示す例では、データベースバッファ１２４は、「トランザクションID=40」で識別されるトランザクションは、更新元レコードID「R2」で識別される更新元レコードを更新して、生成レコードID「R6」で識別される生成レコードが作成されたことを示す。また、「トランザクションID=50」で識別されるトランザクションは、生成レコードID「R7」で識別される生成レコードを追加したことを示す。「トランザクションID=50」で識別されるトランザクションは、更新元レコードID「R2」で識別される更新元レコードを更新して、生成レコードID「R8」で識別される生成レコードが作成されたことを示す。同様に、「トランザクションID=50」で識別されるトランザクションは、更新元レコードID「R4」で識別される更新元レコードを削除して、生成レコードID「R9」で識別される生成レコードが作成されたことを示す。

【００３９】

ログＤＢ１２５は、トランザクションの更新履歴情報を記憶する。この情報は、トランザクションのロールバック時に更新データを無効化する場合や、システムがダウンした場合にデータベースを最新状態にリカバリするために使用される。

【００４０】
制御部１３０は、制御プログラム、各種の処理手順などを規定したプログラムおよび所要データを格納するための内部メモリを有する。制御部１３０は、定義受付け部１３１と定義情報管理部１３２と受付け部１３３とデータ操作部１３４とレコード操作部１３５とバッファ管理部１３６とトランザクション管理部１３７と更新確定部１３８とログ管理部１３９とを有する。例えば、制御部１３０は、ＡＳＩＣ（Application Specific Integrated Circuit）やＦＰＧＡ（Field Programmable Gate Array）などの集積回路、又は、ＣＰＵ（Central Processing Unit）やＭＰＵ（Micro Processing Unit）などの電子回路である。
【００４１】
定義受付け部１３１は、データベース資源の定義を利用者から受付ける。また、定義受付け部１３１は、データベース資源の定義として、データ毎の更新競合時の属性をカラムごとに指定した確定手順情報を受付ける。例えば、定義受付け部１３１は、更新競合時の属性として、「Merge」、「OverWrite」、「Max」、「Min」、「Error」、「Function」を利用者から受付ける。
【００４２】
定義情報管理部１３２は、定義受付け部１３１によって利用者から受付けたデータベースの定義情報を確定手順情報ＤＢ１２１に格納させる。また、定義情報管理部１３２は、確定手順情報ＤＢ１２１から確定手順情報を読み出し、読み出した確定手順情報を確定手順情報ＤＢバッファ１２２へ展開する。例えば、定義情報管理部１３２は、確定手順情報ＤＢ１２１が記憶する図３に示した情報を読み出し、図４に示すように確定手順情報ＤＢバッファ１２２へ展開する。
【００４３】
受付け部１３３は、利用者端末２００からデータベース操作を受付ける。例えば、受付け部１３３は、データベースの参照、更新、トランザクションの開始及び終了を受付け、受付けた内容をデータ操作部１３４へ転送する。また、受付け部１３３は、利用者端末２００からトランザクションのコミット処理を受付けた場合、データ操作部１３４へ通知する。
【００４４】
データ操作部１３４は、受付け部１３３を介して利用者端末２００から受付けたデータベース操作をデータ操作命令として解釈する。例えば、データ操作部１３４は、受付け部１３３からトランザクションの開始及び終了を受付けた場合、受付けた開始及び終了を解釈してトランザクション管理部１３７へ通知する。
【００４５】
また、データ操作部１３４は、利用者端末２００から受付けたトランザクションのアクセス手順を生成する。例えば、データ操作部１３４は、受付け部１３３が受付けたデータ操作命令(ＳＱＬ)と定義情報を参照し、データベース１２３へのアクセス手順を生成する。図７Ａ～図７Ｃを用いて、データ操作部１３４によるデータベースへのアクセス決定を説明する。
【００４６】
例えば、図７Ａに示すＳＱＬ文は、利用者端末から入力されたもので、データ操作部１３４がデータベース１２３に記憶されるテーブル「TBL1」のレコードとして「100、A、Init、5000、・・・」を追加するようにデータベースをアクセスする手順を作成する。
【００４７】
また、図７Ｂに示すＳＱＬ文は、データ操作部１３４がデータベース１２３に記憶されるレコード「100、A、Init、5000、・・・」のカラム２の値を「B」、カラム４の値を「カラム4の値-1000」に更新するようにデータベースにアクセスする手順を作成する。
【００４８】
また、図７Ｃに示すＳＱＬ文は、データ操作部１３４がデータベース１２３に記憶されるレコード「100、A、Init、5000、・・・」のカラム３の値を「Upd」、カラム４の値を

「カラム4の値-1000」に更新するようデータベースをアクセスする手順を作成する。

【００４９】

また、データ操作部１３４は、生成したアクセス手順をレコード操作部１３５へ転送し、データの検索や更新を実行させる。ここで、データ操作部１３４は、トランザクションが実行される独立性水準を記憶し、データ検索時にはトランザクションの開始時刻又はデータ操作命令の実行時刻のどちらのコミット済みデータを検索するのかをレコード操作部１３５へ通知する。

【００５０】

レコード操作部１３５は、データベースの検索や更新、トランザクションの終了、ログの記録を実行する。なお、レコード操作部１３５は、実施例１にかかるデータ生成部１３の一例である。すなわち、レコード操作部１３５は、データを更新するトランザクションを受付けた場合に、トランザクションを実行して、データから確定前の更新データを生成する。

【００５１】

レコード操作部１３５は、データ操作部１３４からデータベースの検索を受付けた場合、バッファ管理部１３６を呼び出して、データベース１２３のレコードをデータベースバッファ１２４に展開する。また、レコード操作部１３５は、データ操作部１３４によって指示された手順に従い、データベース１２３を検索してレコードを返却する。

【００５２】

また、レコード操作部１３５は、データ操作部１３４からＳＱＬ文を受付けた場合、データの更新を実行する。ここで、レコード操作部１３５は、更新の度に新たなレコードを作成し、レコード参照時には資源を排他しないので、たとえ同一のレコードを更新する場合でも、参照処理を排他待ちさせずに処理する。また、レコード操作部１３５は、バッファ管理部１３６に通知し、データベースバッファ１２４を取得する。

【００５３】

例えば、レコード操作部１３５は、データを追加する場合、追加するレコードをデータベースバッファ１２４上に追加する。また、レコード操作部１３５は、更新フラグに「y」を格納することでレコード中のデータを更新したカラムを識別させる。上述した図５及び図７Ａを例にして説明する。レコード操作部１３５は、データ操作部１３４から図７Ａに示したＳＱＬ文を受付けた場合、図５の「レコードID=R2」で識別されるレコードを生成する。

【００５４】

また、例えば、レコード操作部１３５は、データを更新する場合、データベースバッファ１２４上に展開された更新元のレコードをコピーし、コピーしたレコードを更新する。ここで、レコード操作部１３５は、レコード内の更新フラグに「y」を格納することでレコード中のデータを更新したカラムを識別させる。

【００５５】

上述した図５、図７Ｂ及び図７Ｃを例にして説明する。レコード操作部１３５は、データ操作部１３４から図７Ｂに示したＳＱＬ文のアクセス手順を受付けた場合、図５の「レコードID=R6」で識別されるレコードを生成する。同様に、レコード操作部１３５は、データ操作部１３４から図７Ｃに示したＳＱＬ文を受付けた場合、図５の「レコードID=R8」で識別されるレコードを生成する。

【００５６】

また、レコード操作部１３５は、データを削除する場合、直接データを削除せずに、データベースバッファ１２４上に削除用のレコードを生成する。上述した図５を例にして説明する。レコード操作部１３５は、「レコードID=R4」で識別されるレコードを削除する場合、削除用のレコードとして図５に示した「レコードID=R9」で識別されるレコードを生成する。

【００５７】

また、レコード操作部１３５は、トランザクションで更新したレコードセットとして、

更新、削除する元になった更新元レコードと、更新や削除、追加によって生成した生成レコードとをデータベースバッファ１２４上に記憶させる。上述した図６を例にして説明する。レコード操作部１３５は、更新や削除する元になったレコードと、更新や削除追加により生成したレコードとのアドレスを図６に示したレコードセットとしてデータベースバッファ１２４上に記憶させる。

【００５８】

また、レコード操作部１３５は、更新前のレコードイメージと更新後のレコードイメージをログ管理部１３９に通知して、ログＤＢ１２５にログを記録する。

【００５９】

図２に戻り、バッファ管理部１３６は、データベースバッファ１２４上にデータが展開されていない場合、データベース１２３に格納されたデータを読み出し、データベースバッファ１２４へ展開する。例えば、バッファ管理部１３６は、「レコードID=R2」で識別されるレコードを変更して「レコードID=R6」で識別されるレコードを生成する場合、データベース１２３から「レコードID=R2」を読み出し、読み出したレコードをデータベースバッファ１２４に展開する。

【００６０】

トランザクション管理部１３７は、トランザクションを統合的に管理する。例えば、トランザクション管理部１３７は、トランザクションをコミットする場合には、更新確定部１３８を呼び出し、他に同時走行していたトランザクションの最新の更新結果と、コミット対象のトランザクションの更新結果を、定義しておいた更新競合時の属性に従ってマージさせる。また、トランザクション管理部１３７は、トランザクションがロールバックする場合には、トランザクションで更新した結果を更新前のデータに復元する。

【００６１】

更新確定部１３８は、トランザクション管理部１３７から通知を受付けた場合、レコード操作部１３５が生成した追加レコードの確定や更新レコードの確定を実行して、トランザクションをコミットする。ここでは、更新確定部１３８が追加レコードの確定を実行する場合と更新レコードの確定を実行する場合についてそれぞれ詳細に説明する。

【００６２】

また、更新確定部１３８は、実施例１にかかる判定部１４及び確定部１５の一例である。すなわち、更新確定部１３８は、レコード操作部１３５によって実行されたトランザクションをコミットさせる場合に、データを更新元データとする確定済みデータを生成したコミット済みのトランザクションが存在するか否かを判定する。また、更新確定部１３８は、コミット済みのトランザクションが存在すると判定した場合に、確定済みデータと確定手順情報とに従って確定前の更新データを再び更新して、トランザクションをコミットさせる。

【００６３】

［追加レコードの確定を実行する場合］

更新確定部１３８が追加レコードの確定を実行する場合について説明する。更新確定部１３８は、レコード操作部１３５によって生成されたレコードに更新元レコードが存在しなければ、生成されたレコードを追加レコードであると判定する。続いて、更新確定部１３８は、生成レコードが追加レコードであると判定した場合、生成レコードにコミット時刻を格納して生成レコードを確定させる。

【００６４】

上述した図５及び図６を例にして説明する。更新確定部１３８は、図６に示すレコードセットの「トランザクションID=50」で識別されるトランザクションにおいて、「レコードID=R7」で識別されるレコードには、更新元レコードＩＤが格納されておらず、更新元レコードが存在しないと判定する。すなわち、更新確定部１３８は、「レコードID=R7」で識別されるレコードが追加レコードであると判定する。続いて、更新確定部１３８は、図５に示した「レコードID=R7」で識別されるレコードのコミット時刻に「13:02」を格納する。この結果、更新確定部１３８は、「レコードID=R7」で識別される追加レコードを

確定させる。
【００６５】
［更新レコードの確定を実行する場合］
　また、更新確定部１３８は、生成されたレコードに更新元レコードが存在すれば、生成されたレコードを更新レコードであると判定し、更新レコードの確定を実行する。更新確定部１３８は、更新レコードの確定を実行する場合、同一の更新元レコードを更新するトランザクションが存在するか否かを判定する。
【００６６】
（同一の更新元レコードを更新するトランザクションが存在しない場合）
　更新確定部１３８が同一の更新元レコードを更新するトランザクションが存在しないと判定した場合について説明する。更新確定部１３８は、更新元レコードを読み出し、世代ポインタが格納されているか否かを判定することで同一の更新元レコードを更新するコミット済みのトランザクションが存在するか否かを判定する。ここで、更新確定部１３８は、世代ポインタが格納されていないと判定した場合、同一の更新元レコードを更新するトランザクションが存在しないと判定する。
【００６７】
　続いて、更新確定部１３８は、読み出した更新元レコードを参照世代のレコードとして選択する。また、更新確定部１３８は、選択した参照世代のレコードのコミット処理を排他する。そして、更新確定部１３８は、参照世代のレコードに生成したレコードＩＤと更新者ＩＤとを格納する。また、更新確定部１３８は、生成したレコードにコミット時刻を格納し、更新を確定する。
【００６８】
　上述した図５を用いて説明する。ここで、「トランザクションID=40」で識別されるトランザクションと「トランザクションID=50」で識別されるトランザクションとが同時に処理を開始し、「トランザクションID=40」がコミットした後に「トランザクションID=50」がコミットするものとして説明する。図６のレコードセットにおいて、更新確定部１３８は、「トランザクションID=40」で識別されるトランザクションが「レコードID=R2」で識別されるレコードを更新元にして「レコードID=R6」で識別されるレコードを生成したと判定する。
【００６９】
　更新確定部１３８は、「トランザクションID=40」を確定させる場合、図５に示した「レコードID=R2」の世代ポインタには値が格納されていないと判定する。したがって、更新確定部１３８は、「レコードID=R6」と同一の更新元レコードを更新するコミット済みのトランザクションが存在しないと判定する。続いて、更新確定部１３８は、図８に示すように、「レコードID=R2」の世代ポインタに「R6」、更新者ＩＤに「40」を格納する。そして、更新確定部１３８は、「レコードID=R6」のコミット時刻に「13:00」を格納し、更新を確定する。なお、図８は、更新確定部によるデータの確定を示す図である。
【００７０】
（同一の更新元レコードを更新するトランザクションが存在する場合）
　更新確定部１３８が同一の更新元レコードを更新するトランザクションが存在すると判定した場合について説明する。更新確定部１３８は、読み出した更新元レコードに世代ポインタが格納されていると判定した場合、同一の更新元レコードを更新するトランザクションが存在すると判定する。かかる場合、更新確定部１３８は、参照世代を選択し、選択した参照世代と、確定手順情報ＤＢバッファ１２２に記憶された確定手順情報とに従って生成したレコードに格納された値を確定させる。
【００７１】
（参照世代の選択）
　まず、更新確定部１３８による参照世代の選択について説明する。更新確定部１３８は、読み出した更新元レコードの「世代ポインタ」に格納されているレコードＩＤを抽出し、抽出したレコードＩＤに該当するレコードを読み出す。ここで、抽出されたレコードＩ

Ｄを「次の世代のレコード」と定義する。

【００７２】

そして、更新確定部１３８は、次の世代のレコードの「世代ポインタ」が格納されているか否かを判定する。更新確定部１３８は、「世代ポインタ」が格納されていないと判定した場合、次の世代のレコードを参照世代のレコードとして選択する。一方、更新確定部１３８は、世代ポインタが格納されていると判定した場合、「世代ポインタ」が格納されていないレコードを読み出すまで上述した次の世代のレコードを抽出する処理を繰り返す。更新確定部１３８は、参照世代を選択した場合、選択した参照世代のコミット処理を排他する。

【００７３】

上述した図８を用いて説明する。例えば、「トランザクションID=40」がコミットした後に、「トランザクションID=50」がコミットする場合を説明する。更新確定部１３８は、「トランザクションID=50」が「レコードID=R2」を更新元にして「レコードID=R8」を生成したと判定する。そして、更新確定部１３８は、「レコードID=R2」の世代ポインタを読み出し、図８に示すように、「R6」が格納されていると判定する。この結果、更新確定部１３８は、同一の更新元レコードを更新するトランザクションが存在すると判定する。

【００７４】

続いて、更新確定部１３８は、「次の世代のレコード」として「レコードID=R6」を読み出し、「世代ポインタ」が格納されているか否かを判定する。ここで、更新確定部１３８は、「レコードID=R6」の「世代ポインタ」には値が格納されていないので、「レコードID=R6」を「レコードID=R8」の参照世代のレコードとして選択する。

【００７５】

（更新レコードの確定）

更新確定部１３８は、選択した参照世代のレコードと確定手順情報ＤＢバッファ１２２に記憶された確定手順情報とに従って生成したレコードに格納された値を確定させる。例えば、更新確定部１３８は、更新フラグに格納された値が「y」であるか「n」であるかを判定し、カラムの値が更新されているか否かを判定する。

【００７６】

ここで、更新確定部１３８は、更新フラグに格納された値が「n」である場合、カラムに格納された値が更新されていないと判定し、参照世代のカラムに格納された値を設定する。一方、更新確定部１３８は、更新フラグに格納された値が「y」である場合、カラムの値が更新されていると判定し、確定手順情報ＤＢバッファ１２２に記憶された確定手順を読み出し、競合時の属性を判定する。

【００７７】

図８に示す例では、更新確定部１３８は、「レコードID=R8」の更新フラグ１及び更新フラグ２に格納された値が「n」であるので、カラム１の値に「100」、カラム２の値に「B」を設定する。一方、更新確定部１３８は、「レコードID=R8」の更新フラグ３及び更新フラグ４に格納された値が「y」であるので、図４に示す確定手順情報ＤＢバッファ１２２に記憶される確定手順情報からカラム毎に設定された競合時の属性を判定する。そして、更新確定部１３８は、判定した競合時の属性に従って、カラム３及びカラム４の値を設定する。

【００７８】

（競合時の属性）

更新確定部１３８は、競合時の属性として、「Merge」、「Max」、「Min」、「OverWrite」、「Error」、「Function」を判定し、カラムの値を設定する。ここで、更新確定部１３８が判定する競合時の属性について説明する。

【００７９】

例えば、更新確定部１３８は、カラムに設定された競合時の属性が「Merge」であると判定し、更新した差分値を参照世代のレコードの値に計算した結果を設定する。例えば、

更新確定部１３８は、図８に示すような「レコードID=R8」の更新フラグ４の値が「y」であるので、カラムの値が更新されていると判定し、確定手順情報ＤＢバッファ１２２に記憶された確定手順を読み出す。ここで、カラム４の競合時の属性は「Merge」に設定されているので、更新確定部１３８は、参照世代である「レコードID=R6」のカラム４の値から差分値を計算した値、「3000」を格納する。

【００８０】

更新確定部１３８は、カラムに設定された競合時の属性が「Max」であると判定した場合、更新した値と参照世代のレコード中の値との大小比較を行い、最大値を設定する。また、更新確定部１３８は、カラムに設定された属性が「Min」であると判定した場合、更新した値と参照世代のレコード中の値との大小比較を行い、最小値を設定する。

【００８１】

更新確定部１３８は、カラムに設定された競合時の属性が「OverWrite」であると判定した場合、参照世代のレコードと更新したレコードのコミット時刻を比較する。ここで、更新確定部１３８は、更新したレコードのコミット時刻の方が古ければ、参照世代のレコードに格納された値を設定する。一方、更新確定部１３８は、参照世代のレコードのコミット時刻の方が古ければ、更新したレコードに格納された値を設定する。なお、かかる場合、更新確定部１３８は、レコード毎に格納されたコミット時刻を判定する。

【００８２】

例えば、更新確定部１３８は、図８に示すような「レコードID=R8」の更新フラグ３の値が「y」であるので、カラムの値が更新されていると判定し、確定手順情報ＤＢバッファ１２２に記憶された確定手順を読み出す。ここで、カラム３の競合時の属性は「OverWrite」に設定されているので、更新確定部１３８は、参照世代である「レコードID=R6」と更新したレコード「レコードID=R8」のコミット時刻を比較する。この場合、更新したレコード「レコードID=R8」のコミット時刻の方が古いので、更新確定部１３８は、カラム３の値に「Upd」格納する。

【００８３】

更新確定部１３８は、カラムに設定された競合時の属性が「Error」であると判定した場合、参照世代のレコードで更新されているか否かを判定する。ここで、更新確定部１３８は、参照世代で更新されていると判定した場合、トランザクションをキャンセルする。

【００８４】

更新確定部１３８は、カラムに設定された競合時の属性が「Function」であると判定した場合、更新元、参照世代、追加レコードを引数に利用者が登録したファンクションを呼び出して、更新結果をマージする。ここで、「Function」は、複数の利用者によって文章データの添削をするように単純にマージすることができない場合に用いられる校正機能である。具体的には、更新確定部１３８は、添削した文章データを追記する場合に、既に他の利用者によってデータが追記されているならば、元のデータの後ろに添削した文章データを追記する。

【００８５】

続いて、更新確定部１３８は、レコードの更新を確定させた場合、生成したレコードＩＤと更新者ＩＤとを参照世代のレコードに格納する。また、更新確定部１３８は、コミット時刻を確定させたレコードに格納する。図８に示す例では、更新確定部１３８は、「レコードID=R8」の更新を確定させた場合、参照世代のレコードである「レコードID=R6」の世代ポインタに「R8」、更新者ＩＤに「50」を格納する。また、更新確定部１３８は、「レコードID=R8」のコミット時刻に「13:07」を格納する。

【００８６】

また、更新確定部１３８は、一連のトランザクションとして実行されるレコードの全てを処理したか否かを判定する。更新確定部１３８は、全てのレコードを処理したと判定した場合、排他を開放し、トランザクションをコミットする。

【００８７】

更新確定部１３８は、トランザクションをコミットする時にデータベースに加えた変更

、すなわち、コミット時刻の記録や世代ポインタの格納、トランザクションが競合した場合の各カラムの更新確定結果をログ管理部１３９へ転送する。

【００８８】

図９を用いてトランザクションをコミットした後のデータベースの状態の一例を説明する。図９は、更新確定部１３８がトランザクションをコミットした後のデータベースの状態の一例を示す図である。図５に示した例と比較し、図９に示すトランザクションをコミットした後のデータベースの「レコードID＝R2」には、更新者ＩＤに「40」、世代ポインタに「R6」が新たに格納される。「レコードID＝R4」には、更新者ＩＤに「50」、世代ポインタに「R9」が新たに格納される。「レコードID＝R6」には、更新者ＩＤに「50」、コミット時刻に「13:00」、世代ポインタに「R8」が新たに格納される。「レコードID＝R7」には、コミット時刻に「13:02」が新たに格納される。「レコードID＝R8」には、コミット時刻に「13:07」が新たに格納される。また、「レコードID＝R8」のカラム２の値が「B」、カラム４の値が「3000」に更新される。「レコードID＝R9」には、コミット時刻に「13:09」が新たに格納される。

【００８９】

ログ管理部１３９は、トランザクションの更新情報をログＤＢ１２５に格納する。例えば、ログ管理部１３９は、レコード操作部１３５によって更新されたレコードイメージを受信し、受信したレコードイメージをログＤＢ１２５に格納する。また、ログ管理部１３９は、更新確定部１３８からトランザクションをコミットした後のデータを受信し、受信したレコードイメージをログＤＢ１２５に格納する。また、ログ管理部１３９は、トランザクション管理部１３７によってロールバックすると判定された場合には、トランザクションで更新した結果を更新前のデータに復元する。

【００９０】

［データベースサーバ１００による処理の処理手順］

次に、図１０及び図１１を用いて、実施例２にかかるデータベースサーバ１００における処理の処理手順を説明する。ここでは、図１０を用いて、実施例２にかかるデータベースサーバ１００の更新確定部１３８によるコミット処理の処理手順を説明し、図１１を用いて、更新確定部１３８によるマージ処理の処理手順を説明する。

【００９１】

（更新確定部１３８によるコミット処理の処理手順）

図１０は、更新確定部１３８による処理の処理手順を示すフローチャートである。図１０を用いて更新確定部１３８によるコミット処理の処理手順を説明する。更新確定部１３８は、受付け部１３３がコミットを受付けた場合（ステップＳ１０１、Ｙｅｓ）、更新レコードセットを一つ読み出す（ステップＳ１０２）。そして、更新確定部１３８は、読み出したレコードの更新元レコードＩＤが存在するか否かを判定する（ステップＳ１０３）。

【００９２】

更新確定部１３８は、更新元レコードＩＤが存在すると判定した場合には（ステップＳ１０３、Ｙｅｓ）、更新元レコードを読み出し、読み出したレコードのコミット処理を排他する（ステップＳ１０４）。続いて、更新確定部１３８は、更新元レコードの世代ポインタが格納されているか否かを判定する（ステップＳ１０５）。

【００９３】

ここで、更新確定部１３８は、世代ポインタが格納されていないと判定した場合（ステップＳ１０５、Ｎｏ）、世代ポインタと更新者ＩＤとを直前の確定済みレコードに格納し（ステップＳ１０９）、ステップＳ１１０の処理を実行する。

【００９４】

一方、更新確定部１３８は、世代ポインタが格納されていると判定した場合（ステップＳ１０５、Ｙｅｓ）、世代ポインタが示すレコードを読み出して、読み出したレコードのコミット処理を排他する（ステップＳ１０６）。続いて、更新確定部１３８は、読み出したレコードに世代ポインタが格納されているか否かを判定する（ステップＳ１０７）。こ

こで、更新確定部１３８は、世代ポインタが格納されていると判定した場合（ステップＳ１０７、Ｙｅｓ）、ステップＳ１０６の処理に戻り、以降の処理を実行する。

【００９５】

一方、更新確定部１３８は、世代ポインタが格納されていないと判定した場合（ステップＳ１０７、Ｎｏ）、読み出したレコードを排他して更新のマージ処理を実行する（ステップＳ１０８）。続いて、更新確定部１３８は、世代ポインタと更新者ＩＤとを直前の確定済みレコードに格納し（ステップＳ１０９）、ステップＳ１１０の処理を実行する。

【００９６】

更新確定部１３８は、更新元レコードＩＤが存在しないと判定した場合には（ステップＳ１０３、Ｎｏ）、生成したレコードにコミット時刻を格納する（ステップＳ１１０）。続いて、更新確定部１３８は、更新レコードセットに格納された全てのレコードの処理を実行したか否かを判定する（ステップＳ１１１）。

【００９７】

ここで、更新確定部１３８は、更新レコードセットに格納された全てのレコードの処理を実行していないと判定した場合（ステップＳ１１１、Ｎｏ）、ステップＳ１０２の処理に戻り、以降の処理を実行する。一方、更新確定部１３８は、更新レコードセットに格納された全てのレコードの処理を実行したと判定した場合（ステップＳ１１１、Ｙｅｓ）、排他を開放し（ステップＳ１１２）、処理を終了する。

【００９８】

（更新確定部１３８によるマージ処理の処理手順）

次に、図１１を用いて更新確定部１３８によるマージ処理の処理手順を説明する。なお、この処理は、図１０のステップＳ１０８に対応する。

【００９９】

更新確定部１３８は、カラムを一つ選択し（ステップＳ２０１）、選択したカラムに格納された情報を更新したか否かを判定する（ステップＳ２０２）。ここで、更新確定部１３８は、選択したカラムに格納された情報を更新していないと判定した場合（ステップＳ２０２、Ｎｏ）、参照世代のレコードに格納された値を選択したカラムに格納し（ステップＳ２０３）する。続いて、更新確定部１３８は、全てのカラムについて処理を実行したか否かを判定する（ステップＳ２１９）。

【０１００】

更新確定部１３８は、選択したカラムに格納された情報を更新したと判定した場合（ステップＳ２０２、Ｙｅｓ）、確定手順情報ＤＢバッファ１２２に記憶された確定手順情報から選択したカラムの更新競合時の属性を判定する（ステップＳ２０４）。

【０１０１】

そして、更新確定部１３８は、属性が「Merge」であるか否かを判定する（ステップＳ２０５）。ここで、更新確定部１３８は、属性が「Merge」であると判定した場合（ステップＳ２０５、Ｙｅｓ）、更新した差分値を参照世代のレコードが格納する値に計算した結果を設定する（ステップＳ２０６）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。

【０１０２】

一方、更新確定部１３８は、属性が「Merge」ではないと判定した場合（ステップＳ２０５、Ｎｏ）、属性が「Max」又は「Min」であるか否かを判定する（ステップＳ２０７）。ここで、更新確定部１３８は、属性が「Max」又は「Min」であると判定した場合（ステップＳ２０７、Ｙｅｓ）、更新した値と参照世代のレコード値とを比較し、最大値又は最小値を設定する（ステップＳ２０８）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。

【０１０３】

一方、更新確定部１３８は、属性が「Max」又は「Min」ではないと判定した場合（ステップＳ２０７、Ｎｏ）、属性が「OverWrite」であるか否かを判定する（ステップＳ２０９）。ここで、更新確定部１３８は、属性が「OverWrite」であると判定した場合（ステ

ップＳ２０９、Ｙｅｓ）、参照世代のコミット時刻よりも読み出したレコードのコミット時刻が古いか否かを判定する（ステップＳ２１０）。

【０１０４】

ここで、更新確定部１３８は、読み出したレコードのコミット時刻よりも参照世代のコミット時刻が古いと判定した場合（ステップＳ２１０、Ｎｏ）、読み出したレコードの値を設定する（ステップＳ２１１）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。一方、更新確定部１３８は、参照世代のコミット時刻よりも読み出したレコードのコミット時刻が古いと判定した場合（ステップＳ２１０、Ｙｅｓ）、参照世代のレコードの値を設定する（ステップＳ２１２）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。

【０１０５】

一方、更新確定部１３８は、属性が「OverWrite」ではないと判定した場合（ステップＳ２０９、Ｎｏ）、属性が「Error」であるか否かを判定する（ステップＳ２１３）。ここで、更新確定部１３８は、属性が「Error」であると判定した場合（ステップＳ２１３、Ｙｅｓ）、参照世代のレコードが更新されているか否かを判定する（ステップＳ２１４）。

【０１０６】

ここで、更新確定部１３８は、参照世代のレコードが更新されていると判定した場合（ステップＳ２１４、Ｙｅｓ）、トランザクションをキャンセルする（ステップＳ２１５）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。一方、更新確定部１３８は、参照世代のレコードが更新されてないと判定した場合（ステップＳ２１４、Ｎｏ）、読み出したレコードで更新した値を設定する（ステップＳ２１６）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。

【０１０７】

一方、更新確定部１３８は、属性が「Error」ではないと判定した場合（ステップＳ２１３、Ｎｏ）、属性が「Function」であると判定する（ステップＳ２１７）。そして、更新確定部１３８は、利用者によって登録されたファンクションを読み出し、更新結果をマージする（ステップＳ２１８）。続いて、更新確定部１３８は、全てのカラムについてマージ処理を実行したか否かを判定する（ステップＳ２１９）。

【０１０８】

更新確定部１３８は、全てのカラムについて処理を実行していないと判定した場合（ステップＳ２１９、Ｎｏ）、ステップＳ２０１に戻り、以降の処理を実行する。一方、更新確定部１３８は、全てのカラムについて処理を実行したと判定した場合（ステップＳ２１９、Ｙｅｓ）、処理を終了する。

【０１０９】

［実施例２の効果］

上述してきたように、本実施例では、レコード操作部１３５がデータベース１２３から読み出したレコードからデータベースバッファ１２４上に新たなデータを生成する。また、更新確定部１３８は、レコード操作部１３５が生成したレコードを確定する場合、同一の更新元レコードから生成したレコードが存在するか否かを判定する。そして、更新確定部１３８は、更新元が同一のレコードが存在する場合に、確定手順ＤＢバッファ１２２に格納された確定手順情報に基づいて、データを確定させることで、競合するトランザクションを同時に実行することができる。

【０１１０】

また、本実施例にかかるデータベースサーバは、データとして複数のデータ項目を有するレコードを記憶し、データ項目ごとにレコードを確定させる確定手順情報を記憶する。したがって、データベースサーバは、データを更新する場合、データ項目ごとにデータを確定させることができるので、レコードの更新処理を効率的に実行できる。

【０１１１】
また、本実施例にかかるデータベースサーバは、レコードを更新した場合、レコードの更新したデータ項目ごとに更新フラグを付与するので、データを確定させる項目を判定することができるので、レコードの更新処理を効率的に実行できる。
【０１１２】
また、本実施例にかかるデータベースサーバは、更新元データからのデータを更新した場合、更新元データに世代ポインタを格納する。したがって、データベースサーバは、世代ポインタを読み出すことで同一の更新元データを更新するトランザクションが存在するか否かを判定することができる。
【０１１３】
また、本実施例にかかるデータベースサーバは、トランザクションのコミット処理を排他するが、更新処理を排他しないので、同一のデータを更新するトランザクションが存在した場合でも、トランザクションの排他待ちをしなくてもよい。したがって、データベースサーバは、更新競合時のレスポンスやスループットを向上させることができる。
【０１１４】
また、本実施例にかかるデータベースサーバは、アプリケーションとの対話処理にも適用できるので、アプリケーション設計者がトランザクションの排他を考慮してアプリケーションを設計しなくてもよい。したがって、データベースサーバは、アプリケーション設計者の負担を軽減することができる。
【０１１５】
また、本実施例にかかるデータベースサーバは、データベースのレプリケーションによって、主系ＤＢと従系ＤＢを用いる場合においても、更新ログをお互いに送付して反映することで、両系での更新を実現することが出来る。
【０１１６】
また、本願の開示するデータベースサーバは、排他制御やデータキャッシュを最新化させる同期処理を削減することができる。このため、従来クラウド環境を構築する場合に問題となった、頻繁な通信による同期処理を省略することができるので、スケールアウトの限界を克服できる。したがって、本願の開示するデータベースサーバは、クラウド環境下に適した大規模にスケールできるデータベースを提供する技術の一部として活用することができる。
【実施例３】
【０１１７】
ところで、本願の開示するデータベースサーバは、上述した実施例以外にも、種々の異なる形態にて実施されてよい。そこで、実施例３では、本願の開示するデータベースサーバの他の実施例について説明する。
【０１１８】
（システム構成等）
本実施例において説明した各処理のうち自動的に行われるものとして説明した処理の全部または一部を手動的に行うこともできる。あるいは、手動的に行われるものとして説明した処理の全部又は一部を公知の方法で自動的に行うこともできる。この他、上記文章中や図面中で示した処理手順、制御手順、具体的名称については、特記する場合を除いて任意に変更することができる。
【０１１９】
データベースサーバは、利用者端末からデータの検索、更新を受付けるものとして説明したが、これに限定されるものではない。例えば、データベースサーバは、利用アプリケーションサーバであってもよい。また、データベースサーバと利用者端末とが１対１で接続されるものとして説明したが、これに限定されるものではない。例えば、データベースサーバは、複数の利用者端末と接続されてもよい。
【０１２０】
ログ管理部１３９がログの採取を実行し、ログＤＢ１２５へ記憶させるものとして説明

したがこれに限定されるものではない。例えば、ログを記録するかわりに、トランザクション終了時にデータベースバッファの内容をデータベースに同期して永続化することで、ログを記録せずにデータベースの永続性を保証するようにしても良い。

【０１２１】

データ操作部１３４は、データの検索時にトランザクションの開始時刻又はデータ操作命令の実行時刻のどちらのコミット済みデータを検索するのかをレコード操作部１３５へ通知するものとして説明したが、これに限定されない。例えば、データ操作部１３４は、検索する時刻についての情報をアクセス手順と共にＳＱＬ文に格納し、レコード操作部１３５に転送してもよい。

【０１２２】

また、データの検索時に、レコード操作部１３５は、コミット時刻として、トランザクションの終了時の時刻を設定するように説明したがこれに限定されるものではない。例えば、レコード操作部１３５は、トランザクションの開始時刻でデータを検索してもよい。

【０１２３】

また、レコード操作部１３５は、従来のＭＶＣＣ（MultiVersion Concurrency Control）のトランザクションにとって有効な世代か無効な世代かを判定する読み取り世代の判定条件を変更することで、同一レコードを更新するトランザクションを同時に走行させる。レコード操作部１３５は、従来の判定条件の一つ「レコードの更新者ＩＤが未設定である」を「レコードの更新者ＩＤが未設定でかつ、更新元のレコードではない」に変更する。

【０１２４】

図１２は、レコード操作部１３５によって読み取り可能と判定されたレコードＩＤをトランザクション毎に示す図である。ここで、説明の便宜上、スナップショット取得時において、次のトランザクションに割り当てられるトランザクションＩＤは１００と仮定する。図１２に示すように、例えば、上述した図６のレコードセットがスナップショット中であった場合、レコード操作部１３５は、「トランザクションID=40」で識別されるトランザクションについて、「レコードID=R1、R4、R5、R6」を読み取り可能と判定する。また、レコード操作部１３５は、「トランザクションID=50」で識別されるトランザクションについて、「レコードID=R1、R5、R7、R8、R9」を読み取り可能と判定する。

【０１２５】

更新確定部１３８は、カラムを一つ選択し選択した順序でマージ処理を実行するものとして説明したが、これに限定されない。例えば、更新確定部１３８は、複数のカラムのマージ処理を並列して実行しても良い。さらに、更新確定部１３８は、カラムを任意の選択順序で選択することが可能である。また、図１１に示したステップＳ２０５、ステップＳ２０７、ステップＳ２０９、ステップＳ２１３の処理は、どちらを先に実行してもよい。

【０１２６】

また、図示した記憶部が格納する情報は一例に過ぎず、必ずしも図示のごとく情報が格納される必要はない。例えば、データベース１２３は、トランザクションの開始時刻を記憶するようにしてもよい。

【０１２７】

また、図示した各構成部は、機能概念的なものであり、必ずしも物理的に図示のごとく構成されていることを要しない。例えば、データベースサーバ１００は、データ操作部１３４とレコード操作部１３５とは統合されてもよい。さらに、各装置にて行われる各処理機能は、その全部または任意の一部が、ＣＰＵおよび当該ＣＰＵにて解析実行されるプログラムにて実現され、あるいは、ワイヤードロジックによるハードウェアとして実現され得る。

【０１２８】

（プログラム）

ところで、上記実施例で説明した各種の処理は、あらかじめ用意されたプログラムをパーソナルコンピュータやワークステーションなどのコンピュータシステムで実行すること

によって実現することができる。そこで、以下では、上記実施例と同様の機能を有するプログラムを実行するコンピュータシステムの一例を説明する。

【０１２９】

図１３は、データベース更新プログラムを実行するコンピュータシステムを示す図である。図１３に示すように、コンピュータシステム３００は、バス３１０とネットワークインターフェース３２０とＨＤＤ３３０とＲＡＭ３４０とＣＰＵ３５０とＲＯＭ３６０とを有する。ここで、ネットワークインターフェース３２０は、図２に示した通信制御Ｉ／Ｆ部１１０に対応する。

【０１３０】

また、ＲＯＭ３６０には、上記実施例と同様の機能を発揮するプログラムが予め記憶されている。つまり、図１３に示すように、ＲＯＭ３６０は、データ生成プログラム３６１とデータ判定プログラム３６２とデータ確定プログラム３６３とが予め記憶されている。

【０１３１】

そして、ＣＰＵ３５０は、データ生成プログラム３６１とデータ判定プログラム３６２とデータ確定プログラム３６３とを読み出してＲＡＭ３４０に展開する。そして、ＣＰＵ３５０は、データ生成プログラム３６１をデータ生成プロセス３５１として実行する。またＣＰＵ３５０は、データ判定プログラム３６２をデータ判定プロセス３５２として実行する。またＣＰＵ３５０は、データ確定プログラム３６３をデータ確定プロセス３５３として実行する。なお、データ生成プロセス３５１は、図２に示したレコード操作部１３５に対応する。また、データ判定プロセス３５２及びデータ確定プロセス３５３は、更新確定部１３８に対応する。また、ＨＤＤ３３０には、図２に示した確定手順情報ＤＢ１２１に対応する確定手順情報テーブル３３１が設けられる。

【０１３２】

ところで、上記したプログラム３６１～３６３は、必ずしもＲＯＭ３６０に記憶させておく必要はない。例えば、コンピュータシステム３００に挿入されるフレキシブルディスク（ＦＤ）、ＣＤ－ＲＯＭ、ＭＯディスク、ＤＶＤディスク、光磁気ディスク、ＩＣカードなどの「可搬用の物理媒体」に記憶させておくようにしてもよい。また、コンピュータシステム３００の内外に備えられるハードディスクドライブ（ＨＤＤ）などの「固定用の物理媒体」に記憶させておいてもよい。さらに、公衆回線、インターネット、ＬＡＮ（Local Area Network）、ＷＡＮ（Wide Area Network）などを介してコンピュータシステム３００に接続される「他のコンピュータシステム」に記憶させておいてもよい。そして、コンピュータシステム３００がこれらからプログラムを読み出して実行するようにしてもよい。

【０１３３】

すなわち、このプログラムは、上記した「可搬用の物理媒体」、「固定用の物理媒体」、「通信媒体」などの記録媒体に、コンピュータ読み取り可能に記憶されるものである。そして、コンピュータシステム３００は、このような記録媒体からプログラムを読み出して実行することで上記した実施例と同様の機能を実現する。なお、この他の実施例でいうプログラムは、コンピュータシステム３００によって実行されることに限定されるものではない。例えば、他のコンピュータシステムまたはサーバがプログラムを実行する場合や、これらが協働してプログラムを実行するような場合にも、本発明を同様に適用することができる。

【符号の説明】

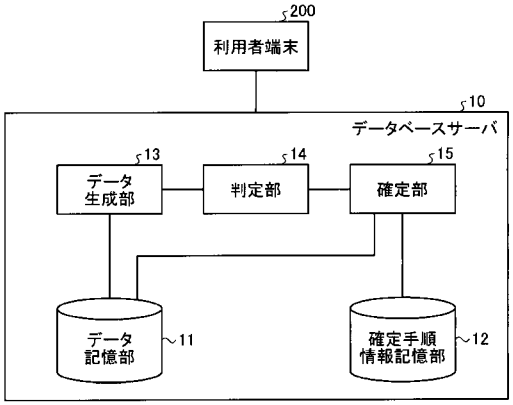
【０１３４】

１０　データベースサーバ
１１　データ記憶部
１２　確定手順情報記憶部
１３　データ生成部
１４　判定部
１５　確定部

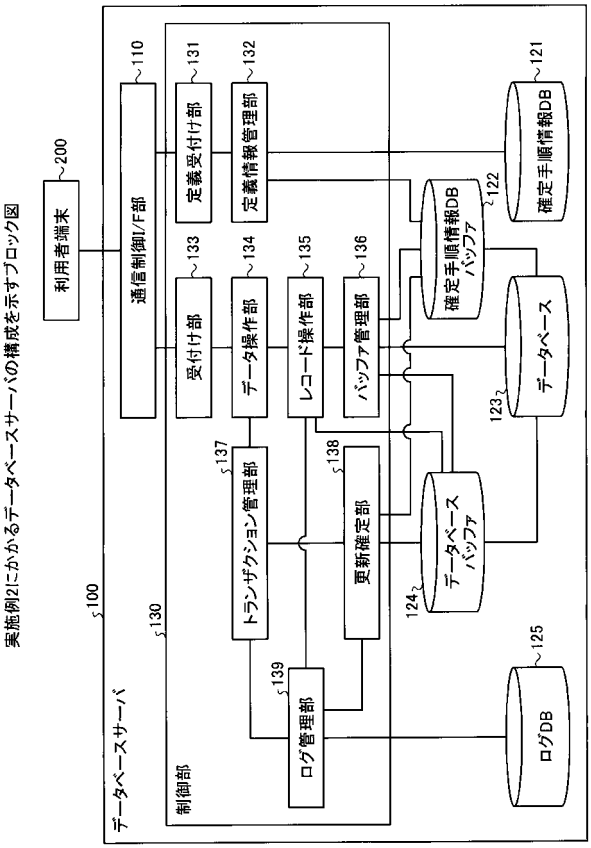
１００　データベースサーバ
１１０　通信制御Ｉ／Ｆ部
１２１　確定手順ＤＢ
１２２　確定手順ＤＢバッファ
１２３　データベース
１２４　データベースバッファ
１２５　ログＤＢ
１３０　制御部
１３１　定義受付け部
１３２　定義情報管理部
１３３　受付け部
１３４　データ操作部
１３５　レコード操作部
１３６　バッファ管理部
１３７　トランザクション管理部
１３８　更新確定部
１３９　ログ管理部
２００　利用者端末
３００　コンピュータシステム
３１０　バス
３２０　ネットワークインターフェース
３３０　ＨＤＤ
３３１　確定手順情報テーブル
３４０　ＲＡＭ
３５０　ＣＰＵ
３５１　データ生成プロセス
３５２　データ判定プロセス
３５３　データ確定プロセス
３６０　ＲＯＭ
３６１　データ生成プログラム
３６２　データ判定プログラム
３６３　データ確定プログラム

【図１】

実施例1にかかるデータベースサーバの構成を示すブロック図



【図２】



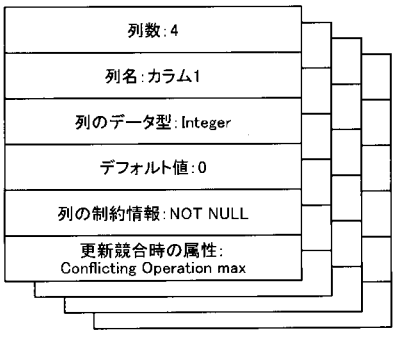
【図３】

確定手順情報DBが記憶する情報の一例を示す図

列名	列のデータ型	デフォルト値	列の制約情報	更新競合時の属性
カラム1	Integer	0	NOT NULL	Conflicting Operation Max
カラム2	Char (1)	-	-	Conflicting Operation OverWrite
カラム3	Char (5)	-	-	Conflicting Operation OverWrite
カラム4	Integer	-	-	Conflicting Operation Merge

【図４】

確定手順情報DBバッファが記憶する情報の一例を示す図



【図５】

データベースがテーブル「TBL1」として記憶する情報の一例を示す図

レコードID	生成者ID	更新者ID	コミット時刻	世代ポインタ	更新フラグ1	カラム1	更新フラグ2	カラム2	更新フラグ3	カラム3	更新フラグ4	カラム4
R2	10	-	10:00	-	y	100	y	A	y	Init	y	5000
R4	10	-	10:00	-	y	30	y	F	y	Init	y	3200
R6	40	-	-	-	n	100	y	B	n	Init	y	4000
R7	50	-	-	-	y	20	y	C	y	Init	y	450
R8	50	-	-	-	n	100	n	A	y	Upd	y	4000
R9	50	50	-	-	y	-	y	-	y	-	y	-

【図６】

データベースバッファが記憶するトランザクションのレコードセットの一例を示す図

トランザクションID	更新元レコードID	生成レコードID	処理
40	R2	R6	更新
50	–	R7	追加
50	R2	R8	更新
50	R4	R9	削除

【図７Ａ】

データ操作部がアクセス手順として生成したSQL文の一例を示す図

```
Insert into SCM1. TBL1 Values
(100, A, Init, 5000, ･･･)
Commit
```

【図７Ｂ】

データ操作部がアクセス手順として生成したSQL文の別例を示す図

```
Update SCM1. TBL1
Set Column2='B', Column4=Column4-1000
WHERE Column1=100
```

【図７Ｃ】

データ操作部がアクセス手順として生成したSQL文の別例を示す図

```
Update SCM1. TBL1
Set Column3='Upd', Column4=Column4-1000
WHERE Column1=100
```

【図８】

更新確定部によるデータの確定を示す図

レコードID	生成者ID	更新者ID	コミット時刻	世代ポインタ	更新フラグ1	カラム1	更新フラグ2	カラム2	更新フラグ3	カラム3	更新フラグ4	カラム4
R2	10	40	10:00	R6	y	100	y	A	y	Init	y	5000
R6	40	50	13:00	R8	n	100	y	B	n	Init	y	4000
R8	50	–	13:07	–	n	100	n	A→B	y	Upd	y	4000→3000

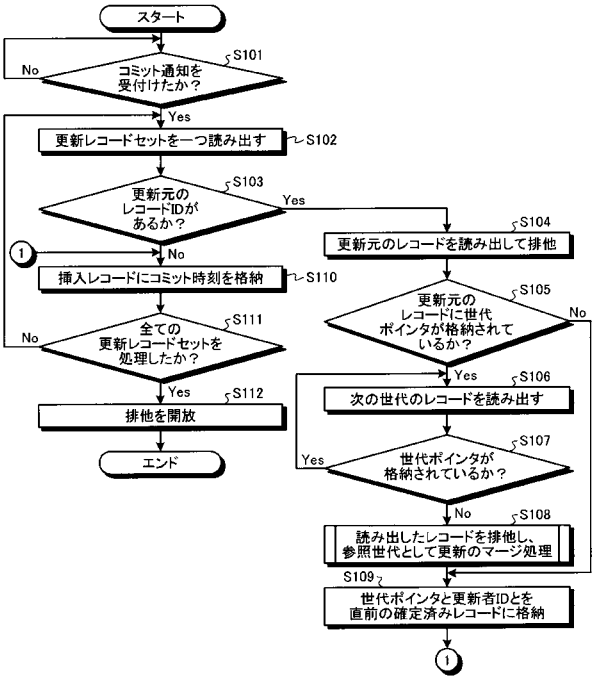
【図９】

更新確定部がトランザクションをコミットした後のデータベースの状態の一例を示す図

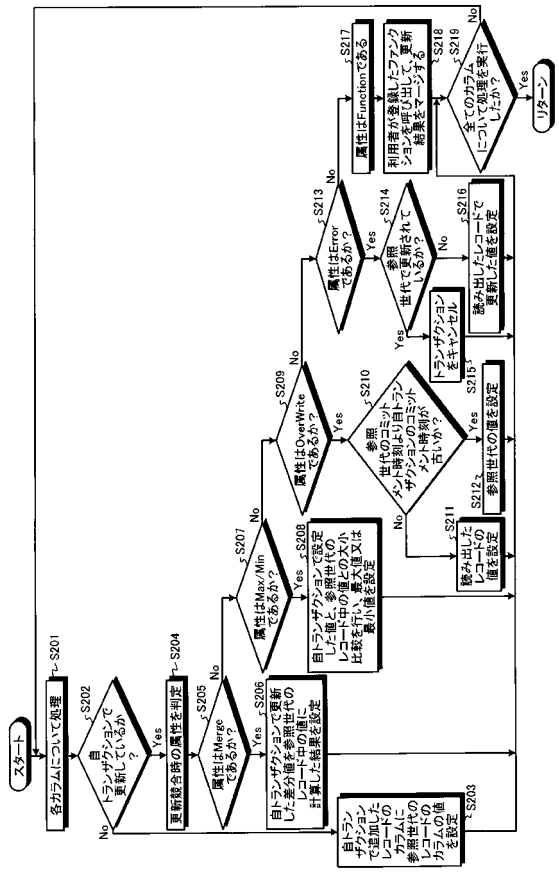
レコードID	生成者ID	更新者ID	コミット時刻	世代ポインタ	更新フラグ1	カラム1	更新フラグ2	カラム2	更新フラグ3	カラム3	更新フラグ4	カラム4
R2	10	40	10:00	R6	y	100	y	A	y	Init	y	5000
R4	10	50	10:00	R9	y	30	y	F	y	Init	y	3200
R6	40	50	13:00	R8	n	100	y	B	n	Init	y	4000
R7	50	–	13:02	–	y	20	y	C	y	Init	y	450
R8	50	–	13:07	–	n	100	n	B	y	Upd	y	3000
R9	50	50	13:09	–	y	–	y	–	y	–	y	–

【図１０】

更新確定部による処理の処理手順を示すフローチャート



【図１１】



【図１２】

レコード操作部によって読み取り可能と判定されたレコードIDをトランザクション毎に示す図

レコードID	生成者ID	更新者ID	ID:40 Visible	ID:50 Visible
R1	10		visible	visible
R2	10			
R3	10	80		
R4	20		visible	
R5	30	110	visible	visible
R6	40		visible	
R7	50			visible
R8	50			visible
R9	50			visible
R10	110			
R11	110			

【図１３】