(19)日本国特許庁(JP)　(12)特　許　公　報(B2)　(11)特許番号
特許第5949937号
(P5949937)
(45)発行日　平成28年7月13日(2016.7.13)　(24)登録日　平成28年6月17日(2016.6.17)

(51)Int.Cl.　F I
G06F 12/00 (2006.01)　G06F 12/00 535F
G06F 12/00 518A

請求項の数 10　(全 24 頁)

(21)出願番号　特願2014-544217(P2014-544217)
(86)(22)出願日　平成25年8月14日(2013.8.14)
(86)国際出願番号　PCT/JP2013/004860
(87)国際公開番号　WO2014/068820
(87)国際公開日　平成26年5月8日(2014.5.8)
審査請求日　平成27年4月10日(2015.4.10)
(31)優先権主張番号　特願2012-237597(P2012-237597)
(32)優先日　平成24年10月29日(2012.10.29)
(33)優先権主張国　日本国(JP)

(73)特許権者　000004237
日本電気株式会社
東京都港区芝五丁目７番１号
(74)代理人　100109313
弁理士　机　昌彦
(74)代理人　100124154
弁理士　下坂　直樹
(72)発明者　海老山　知生
東京都港区芝五丁目７番１号
日本電気株式会社内

審査官　篠塚　隆

最終頁に続く

(54)【発明の名称】トランザクションシステム

(57)【特許請求の範囲】
【請求項１】
キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、
複数の前記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と、
前記データグループを参照および更新するトランザクションのコミット時、前記ＷＡＬ記憶部に記憶された更新履歴に基づいて、前記レコード単位で排他違反の有無を判定し、排他違反がなければ前記トランザクションを成功させ、その更新履歴を前記ＷＡＬ記憶部に記録するトランザクション処理部と、
前記ＷＡＬ記憶部に記憶されている前記更新履歴中のレコードの更新内容を前記レコードデータ記憶部に反映するＷＡＬ反映部と
を有するトランザクションシステム。
【請求項２】
前記ＷＡＬ記憶部は、キーとバリューの組で表現される更新ログを記憶し、前記キーは複数の前記レコードを集約したデータであるデータグループを一意に識別する値であり、前記バリューは前記データグループの更新履歴を表し、また、前記更新履歴は、１以上の部分更新履歴を有し、１つの前記部分更新履歴は、前記データグループを構成する前記レコードの更新内容と更新の順番を表すタイムスタンプとを有し、
前記トランザクション処理部は、
アプリケーションプログラムから入力されたトランザクションで使用する前記データグ

ループを構成する前記レコードの最新の値を前記ＷＡＬ記憶部および前記レコードデータ記憶部から取得し、また、当該データグループに対応する前記更新ログ中の最新の前記タイムスタンプをスタート時タイムスタンプとして前記ＷＡＬ記憶部から取得し、前記トランザクションを実行するＷＡＬ管理部と、

前記トランザクションのコミット時、前記トランザクションのスタート時に取得された前記スタート時タイムスタンプと、前記ＷＡＬ記憶部に現に記憶されている前記データグループの最新の前記タイムスタンプであるコミット時タイムスタンプとを比較し、前記スタート時タイムスタンプと前記コミット時タイムスタンプとが相違するときは、前記スタート時タイムスタンプより後のタイムスタンプを有する前記部分更新履歴中に、前記トランザクションが使用した前記レコードと一致するレコードが存在するか否かを判定し、前記比較と前記判定の結果とに基づいて、前記トランザクションを成功させるか、失敗させるかを決定し、成功させると決定した場合、前記トランザクションで更新した前記レコードの更新内容と前記コミット時タイムスタンプより後のタイムスタンプとを有する新たな部分更新履歴を、前記データグループに対応して前記ＷＡＬ記憶部に現に記憶されている前記更新ログに追加する排他チェック処理部と

を有する請求項１に記載のトランザクションシステム。

【請求項３】

前記ＷＡＬ反映部は、前記レコードデータ記憶部に反映した前記更新ログの複数の部分更新履歴のうち、タイムスタンプの新しい上位所定個数の部分更新履歴は削除せずに残しておく

請求項２に記載のトランザクションシステム。

【請求項４】

前記ＷＡＬ反映部は、削除せずに残しておいた反映済の前記部分更新履歴を識別する情報を前記更新履歴に記録する

請求項３に記載のトランザクションシステム。

【請求項５】

前記排他チェック処理部は、前記判定では、前記スタート時タイムスタンプより後のタイムスタンプを有する前記部分更新履歴中に、前記トランザクションが参照または更新した前記レコードと一致するレコードが存在するか否かを判定する

請求項２乃至４の何れかに記載のトランザクションシステム。

【請求項６】

前記排他チェック処理部は、前記判定では、前記スタート時タイムスタンプより後のタイムスタンプを有する前記部分更新履歴中に、前記トランザクションが更新した前記レコードと一致するレコードが存在するか否かを判定する

請求項２乃至４の何れかに記載のトランザクションシステム。

【請求項７】

前記ＷＡＬ反映部は、トランザクションシステムを構成するＣＰＵのアイドル状態と前記ＷＡＬ記憶部を構成する記憶装置の使用量とに基づいて、前記反映のタイミングを決定する

請求項１乃至６の何れかに記載のトランザクションシステム。

【請求項８】

キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、

複数の前記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と、

トランザクション処理部と、

ＷＡＬ反映部と

を有するトランザクションシステムの制御方法であって、

前記トランザクション処理部が、前記データグループを参照および更新するトランザクションのコミット時、前記ＷＡＬ記憶部に記憶された更新履歴に基づいて、前記レコード単位で排他違反の有無を判定し、排他違反がなければ前記トランザクションを成功させ、

その更新履歴を前記ＷＡＬ記憶部に記録し、
　前記ＷＡＬ反映部が、前記ＷＡＬ記憶部に記憶されている前記更新履歴中のレコードの更新内容を前記レコードデータ記憶部に反映する
トランザクションシステム制御方法。
【請求項９】
　前記ＷＡＬ記憶部は、キーとバリューの組で表現される更新ログを記憶し、前記キーは複数の前記レコードを集約したデータであるデータグループを一意に識別する値であり、前記バリューは前記データグループの更新履歴を表し、また、前記更新履歴は、１以上の部分更新履歴を有し、１つの前記部分更新履歴は、前記データグループを構成する前記レコードの更新内容と更新の順番を表すタイムスタンプとを有し、
　前記トランザクション処理部は、
　アプリケーションプログラムから入力されたトランザクションで使用する前記データグループを構成する前記レコードの最新の値を前記ＷＡＬ記憶部および前記レコードデータ記憶部から取得し、また、当該データグループに対応する前記更新ログ中の最新の前記タイムスタンプをスタート時タイムスタンプとして前記ＷＡＬ記憶部から取得し、前記トランザクションを実行し、
　前記トランザクションのコミット時、前記トランザクションのスタート時に取得された前記スタート時タイムスタンプと、前記ＷＡＬ記憶部に現に記憶されている前記データグループの最新の前記タイムスタンプであるコミット時タイムスタンプとを比較し、前記スタート時タイムスタンプと前記コミット時タイムスタンプとが相違するときは、前記スタート時タイムスタンプより後のタイムスタンプを有する前記部分更新履歴中に、前記トランザクションが使用した前記レコードと一致するレコードが存在するか否かを判定し、前記比較と前記判定の結果とに基づいて、前記トランザクションを成功させるか、失敗させるかを決定し、成功させると決定した場合、前記トランザクションで更新した前記レコードの更新内容と前記コミット時タイムスタンプより後のタイムスタンプとを有する新たな部分更新履歴を、前記データグループに対応して前記ＷＡＬ記憶部に現に記憶されている前記更新ログに追加する
請求項８に記載のトランザクションシステム制御方法。
【請求項１０】
　キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、
　複数の前記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と
を有するコンピュータを、
　前記データグループを参照および更新するトランザクションのコミット時、前記ＷＡＬ記憶部に記憶された更新履歴に基づいて、前記レコード単位で排他違反の有無を判定し、排他違反がなければ前記トランザクションを成功させ、その更新履歴を前記ＷＡＬ記憶部に記録するトランザクション処理部と、
　前記ＷＡＬ記憶部に記憶されている前記更新履歴中のレコードの更新内容を前記レコードデータ記憶部に反映するＷＡＬ反映部と
して機能させるためのプログラム。
【発明の詳細な説明】
【技術分野】
【０００１】
　本発明は、各々がキーバリューストアである複数のレコードに対して参照および更新するトランザクションを実行するトランザクションシステム、トランザクションシステム制御方法、およびプログラムに関する。
【背景技術】
【０００２】
　近年、スケーラブルなアプリケーションの実行基盤として、多数のコンピュータを用いるいわゆるクラウドが注目されている。クラウドのインフラストラクチャにおいて、スケ

ーラブルなデータ永続化手段として、キーバリューストア（Ｋｅｙ Ｖａｌｕｅ Ｓｔｏｒｅ（ＫＶＳ））というものが広く知られている。しかしながら、通常のＫＶＳでは１つのレコードに対するトランザクションしか提供できない。

【０００３】

そこで、各々がＫＶＳである複数のレコードに対してトランザクションをサポートするシステムが本発明に関連する第１の関連技術として提案されている（例えば非特許文献１参照）。この第１の関連技術では、ＫＶＳで複数のレコードに対するトランザクションを実現するために、トランザクションで扱われる複数のレコードをグルーピングし、１つのＫｅｙ Ｖａｌｕｅ（ＫＶ）データとしてＫＶＳに書き込むことで、複数レコードに対するトランザクションを実現している。複数のレコードを１つのＫＶデータに集約したデータを、本明細書では、データグループと呼ぶ。

【０００４】

また上記第１の関連技術では、ＫＶＳに対する排他制御の仕組みとして、楽観的排他を使用している。具体的には、レコード（ＫＶデータ）ごとにタイムスタンプが採番されており、あるトランザクションがスタートしたときに当該タイムスタンプを取得し、レコードを更新してコミットする際にタイムスタンプがトランザクションのスタート時から変化していなければ、タイムスタンプを１つインクリメントしてＫＶＳに書き込む。コミット時にタイムスタンプが変化していれば、当該トランザクションがスタートしてからコミットするまでの間に他のトランザクションによって先にレコードが更新されていることになるので、当該トランザクションを失敗させる。楽観的排他では、排他制御情報などの共有資源を使用しないため、スケールアウトが容易になる。

【０００５】

他方、悲観的排他と呼ばれる排他制御の仕組みを採用したトランザクションシステムが本発明に関連する第２の関連技術として提案されている（例えば特許文献１参照）。この第２の関連技術では、トランザクションからのレコード更新要求に従うデータファイルの更新処理に際しては、更新するレコードへの同時アクセスを排他制御機構にて排他制御する。また、そのレコードの更新をバッファ上で行わせると共に、バッファ上でのレコード更新の履歴を生成する。トランザクションの確定時には、生成しておいたトランザクションに対応するレコード単位の更新履歴をジャーナルファイルに記録する。そして、当該ジャーナルファイルに記録した更新履歴に基づき書き込みバッファを通してブロック単位でデータファイルを更新する。

【０００６】

また、マスタサイトのマスタデータとローカルサイトの複製データとの同期を管理する共有データ管理システムにおいて、マスタデータの更新データをローカルサイトの複製データに反映させるタイミングを、更新の緊急度等に基づいて決定することが、本発明に関連する第３の関連技術として提案されている（例えば特許文献２参照）。

【先行技術文献】

【特許文献】

【０００７】

【特許文献１】特開平９－６２５５０号公報
【特許文献２】特開２００１－３５０６６１号公報

【非特許文献】

【０００８】

【非特許文献１】“ビッグデータの活用に最適なスケールアウト型　新データベース「ＩｎｆｏＦｒａｍｅ　Ｒｅｌａｔｉｏｎａｌ　Ｓｔｏｒｅ」、祐成光樹、田村稔、ＮＥＣ技報、Ｖｏｌ.６５ Ｎｏ.２／２０１２.

【発明の概要】

【発明が解決しようとする課題】

【０００９】

上述したようにスケーラブルなデータ永続化手段としてＫＶＳが注目されており、また

ＫＶＳで複数のレコードに対してトランザクションをサポートするシステムが提案されている。しかしながら、上述したＫＶＳに対する楽観的な排他制御の仕組みでは、１つのレコード（ＫＶデータ）単位の排他しか実現できない。複数のレコードを１つのＫＶデータにまとめてトランザクションを実行するシステムにおいては、まとめられた大きな１つのＫＶデータが排他の単位となるため、本来は排他違反にならないようなトランザクション同士も排他違反になってしまう可能性があり、トラザクションの同時実行性が下がり、スループットが下がってしまう問題がある。本発明に関連する第２の関連技術に示される悲観的排他を適用し、大きな１つのＫＶデータにまとめられる前のレコード単位で排他することが考えられるが、悲観的排他ではスケールアウトが困難になり、ＫＶデータを使用する意味がなくなる。

【００１０】

本発明の目的は、上述した課題、すなわち、複数のレコードを１つのＫＶデータにまとめてトランザクションを実行するシステムではトランザクションの同時実行性が低下する、という課題を解決するトランザクションシステムを提供することにある。

【課題を解決するための手段】

【００１１】

本発明の第１の観点に係るトランザクションシステムは、

キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、

複数の上記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と、

上記データグループを参照および更新するトランザクションのコミット時、上記ＷＡＬ記憶部に記憶された更新履歴に基づいて、上記レコード単位で排他違反の有無を判定し、排他違反がなければ上記トランザクションを成功させ、その更新履歴を上記ＷＡＬ記憶部に記録するトランザクション処理部と、

上記ＷＡＬ記憶部に記憶されている上記更新履歴中のレコードの更新内容を上記レコードデータ記憶部に反映するＷＡＬ反映部と

を有する。

【００１２】

また本発明の第２の観点に係るトランザクションシステム制御方法は、

キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、

複数の上記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と、

トランザクション処理部と、

ＷＡＬ反映部と

を有するトランザクションシステムの制御方法であって、

上記トランザクション処理部が、上記データグループを参照および更新するトランザクションのコミット時、上記ＷＡＬ記憶部に記憶された更新履歴に基づいて、上記レコード単位で排他違反の有無を判定し、排他違反がなければ上記トランザクションを成功させ、その更新履歴を上記ＷＡＬ記憶部に記録し、

上記ＷＡＬ反映部が、上記ＷＡＬ記憶部に記憶されている上記更新履歴中のレコードの更新内容を上記レコードデータ記憶部に反映する。

また本発明の第３の観点に係るプログラムは、

キーとバリューの組で表現されるレコードを記憶するレコードデータ記憶部と、

複数の上記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶するＷＡＬ記憶部と

を有するコンピュータを、

上記データグループを参照および更新するトランザクションのコミット時、上記ＷＡＬ記憶部に記憶された更新履歴に基づいて、上記レコード単位で排他違反の有無を判定し、排他違反がなければ上記トランザクションを成功させ、その更新履歴を上記ＷＡＬ記憶部に記録するトランザクション処理部と、

上記ＷＡＬ記憶部に記憶されている上記更新履歴中のレコードの更新内容を上記レコードデータ記憶部に反映するＷＡＬ反映部と
して機能させる。

【発明の効果】

【００１３】

本発明は上述した構成を有するため、複数のレコードを１つのＫＶデータにまとめてトランザクションを実行するシステムにおけるトランザクションの同時実行性を高めることが可能である。

【図面の簡単な説明】

【００１４】

【図１】本発明の第１の実施形態のブロック図である。

【図２】本発明の第１の実施形態におけるトランザクション処理部の動作の一例を示すフローチャートである。

【図３】本発明の第１の実施形態におけるレコードデータおよびデータグループの説明に使用するデータの例を示す図である。

【図４】本発明の第１の実施形態におけるデータグループの例を示す図である。

【図５】本発明の第１の実施形態におけるＷＡＬ記憶手段に記憶されるＷＡＬの説明図である。

【図６】本発明の第１の実施形態における排他チェック処理部で排他違反無しと判断される一例を示す図である。

【図７】本発明の第１の実施形態における排他チェック処理部で排他違反有りと判断される一例を示す図である。

【図８】本発明の第１の実施形態における排他チェック処理部で排他違反無しと判断される他の例を示す図である。

【図９】本発明の第１の実施形態におけるＷＡＬ反映手段の動作の一例を示すフローチャートである。

【図１０】本発明の第１の実施形態におけるＷＡＬ反映手段で反映されるレコードデータの一例を示す図である。

【図１１】本発明の第２の実施形態のブロック図である。

【図１２】本発明の第２の実施形態においてタイムスタンプの新しい上位所定個数のＷＡＬを削除せずに残しておく説明図である。

【図１３】本発明の第３の実施形態のブロック図である。

【図１４】本発明の第３の実施形態におけるＷＡＬ反映手段の動作の一例を示すフローチャートである。

【図１５】本発明の第４の実施形態のブロック図である。

【発明を実施するための形態】

【００１５】

次に本発明の実施の形態について図面を参照して詳細に説明する。

[第１の実施形態]

[構成]

図１を参照すると、本発明の第１の実施形態にかかるトランザクションシステムは、トランザクション処理部１００、ＷＡＬ記憶手段２００、ＷＡＬ反映手段３００、レコードデータ記憶手段４００、システムカタログ５００を備え、トランザクション処理部１００はさらにＷＡＬ管理手段１０１、リード／ライトセット管理手段１０２、排他チェック処理部１０３を備え、排他チェック処理部１０３はさらにＷＡＬデータ分解手段１１３とレコードデータ照合手段１２３を備える。

【００１６】

トランザクション処理部１００は、１以上備えられている。各々のトランザクション処理部１００は、アプリケーション（アプリケーションプログラム）６００から入力されたトランザクションを受け付け、当該トランザクションを処理する手段である。各々のトラ

ンザクション処理部１００は、他のトランザクション処理部１００と並行して動作する。
【００１７】
ＷＡＬ管理手段１０１は、アプリケーション６００から入力されたトランザクションに対して、当該トランザクションが使用するレコードを論理的にまとめたデータグループを作成し、その更新内容をＣＡＳ（Ｃｏｍｐａｒｅ　Ａｎｄ　Ｓｗａｐ）コマンド等を使用して、ＷＡＬとしてＷＡＬ記憶手段２００に対して書き込んだり、読み出したりする手段である。
【００１８】
ここで、ＷＡＬ（Ｗｒｉｔｅ　Ａｈｅａｄ　Ｌｏｇ）について説明する。一般にＷｒｉｔｅ　Ａｈｅａｄ　Ｌｏｇｇｉｎｇとは、データベースに対する操作前の変更内容をログに書き出すことを指し、ＷＡＬとはログそのものを指す。このように、データベースへの変更は、先ずログに記録され、次いで、ログからデータベースへ反映される。
【００１９】
ＷＡＬ記憶手段２００は、ＷＡＬ管理手段１０１によって生成、または更新されたＷＡＬを保存する手段である。ＷＡＬ記憶手段２００は、複数のトランザクション処理部１００に共通に備えられる。本実施形態において、ＷＡＬ記憶手段２００はＫＶＳによって構成されており、ＷＡＬ自体もＫｅｙとＶａｌｕｅからなるＫＶデータとして保存される。ＫＶＳは、例えばＣＡＳコマンドを使用することにより、１つのＫＶデータに対するアトミック性は保証することができるが、複数のレコードを一度のトランザクションでアトミックに処理することはできない。このため、複数のレコードを論理的にまとめたデータグループを作成し、このデータグループに対する更新内容をＷＡＬとしてＫＶＳに書き込むことで複数レコードをトランザクショナルに処理することを可能としている。
【００２０】
リード／ライトセット管理手段１０２は、アプリケーション６００から入力されたトランザクションで参照や更新したレコードの情報などを抽出する手段である。リード／ライトセット管理手段１０２で抽出された参照レコードや更新レコードの情報などは、排他チェック処理部１０３に渡される。
【００２１】
排他チェック処理部１０３は、トランザクションをコミットする際に、当該トランザクションがスタートしてからコミットするまでにＷＡＬ記憶手段２００に追加されたＷＡＬのデータと、リード／ライトセット管理手段１０２から渡される、当該トランザクションで参照や更新したレコードとを比較して、同じレコードが存在する場合はトランザクションを失敗させ、同じレコードが存在しない場合はトランザクションを成功させる。トランザクションが成功した場合は、ＷＡＬ管理手段１０１が、当該トランザクションによる更新内容をＷＡＬとしてＷＡＬ記憶手段２００に書き込む。
【００２２】
ＷＡＬデータ分解手段１１３は、ＷＡＬ記憶手段２００から取得したＷＡＬデータをレコード単位に分解してＷＡＬデータに含まれるレコードデータを取得する手段である。
【００２３】
レコードデータ照合手段１２３は、当該トランザクションで参照や更新したレコードと、ＷＡＬに含まれるレコードデータとを比較して、同じレコードがあるかどうかを調べ、同じレコードが存在する場合はトランザクションを失敗させ、同じレコードが存在しない場合はトランザクションを成功させる手段である。
【００２４】
ＷＡＬ反映手段３００は、ＷＡＬ記憶手段２００に記憶されているＷＡＬに含まれるレコードデータをレコードデータ記憶手段４００に反映し、反映したＷＡＬをＷＡＬ記憶手段２００から削除する。ＷＡＬ反映手段３００が動作するタイミングはタイマーで指定してもよいし、システム管理者等が任意のタイミングで行っても良い。
【００２５】
レコードデータ記憶手段４００は、レコードデータを保存しておく手段である。本実施

形態において、レコードデータ記憶手段４００はＫＶＳによって構成されており、レコードデータはＫｅｙとＶａｌｕｅからなるＫＶデータとして保存される。なお、ＷＡＬ記憶手段２００とレコードデータ記憶手段４００は、図１では別々のＫＶＳとして表現されているが、１つのＫＶＳにまとめても良い。

【００２６】

システムカタログ５００は、レコードデータ記憶手段４００に保存されているデータのメタ情報や、トランザクションを実行する際に作成するデータグループの情報などのデータを保存する手段である。アプリケーション６００からトランザクションが入力された時、ＷＡＬ管理手段１０１はシステムカタログ５００からデータグループに関連する情報を抽出し、抽出された情報を元に当該トランザクションで使用するデータグループを特定する。

【００２７】

[動作]

次に、本実施形態の動作について説明する。本実施形態は図２に示すフローチャートに従って動作を行う。

【００２８】

まず、トランザクション入力のステップＳ００１で、アプリケーション６００が本システムに対してトランザクションを入力する。

【００２９】

次に、ＷＡＬ特定のステップＳ００２で、ＷＡＬ管理手段１０１が、入力されたトランザクションを解析し、システムカタログ５００の情報を用いて当該トランザクションで処理するデータが所属するデータグループを特定し、当該データグループのこれまでの更新内容が保存されているＷＡＬを特定する。本システムにおいて、ＷＡＬとはトランザクションで扱われる複数のレコードを論理的にまとめたデータグループの更新内容の履歴を１つのＫＶデータにまとめたものである。

【００３０】

ここで、本システムにおけるデータグループの概念を説明する。図３に示すような二種類のデータ、すなわち、商品テーブルＴ００１と入札テーブルＴ００２があるとする。このようなデータに対して、以下のようなトランザクションを実行するとする。

「ユーザーが新たに入札を行った際に、商品テーブルと入札テーブルのデータを更新する」

【００３１】

このようなトランザクションを処理する場合、商品テーブルのデータと入札テーブルのデータをアトミックに更新しなければならない。しかし、ＫＶＳは１つのＫＶデータに対するアトミック性しか保証していないため、例えば、商品テーブルのデータを更新したあと、入札テーブルのデータの更新に失敗した場合、商品テーブルのデータだけが更新された状態になってしまい、アトミック性が保証されないという問題がある。

【００３２】

そこで、トラザクションに関連する複数のデータを論理的にまとめて１つのＫＶデータとし、複数のデータ更新を１回のＫＶＳに対する書き込みで実現することでトラザクションのアトミック性を保証する。

【００３３】

データグループを作成するには、まず各データで共通するＫｅｙ（図３の例では商品ＩＤ）を抽出する。図３に示すデータからは、図４に示すような３つのグループ（Ｇ００１、Ｇ００２、Ｇ００３）が論理的にまとめられる。

【００３４】

次に、本システムにおけるＷＡＬの概念を説明する。図３に示すデータに対して以下のような２つのトランザクションが順番に実行されたとする。

最初のトランザクション：「入札テーブルＴ００２のＩＤがｂ１のレコードの入札額を１２０００に更新し、商品テーブルＴ００１の商品ＩＤがｉ１の最高金額を１２０００に更

新する」
二番目のトランザクション：「入札テーブルＴ００２のＩＤがｂ２のレコードの入札額を１３０００に更新し、商品テーブルＴ００１の商品ＩＤがｉ１の最高金額を１３０００に更新する。」
【００３５】
まず、最初のトランザクション１によって商品テーブルＴ００１の１行目のデータと入札テーブルＴ００２の１行目のデータが更新され、二番目のトランザクション２によって商品テーブルＴ００１の１行目のデータと入札テーブルＴ００２の２行目のデータが更新される。これらの２つのトランザクションで更新されるデータは全て同じデータグループに所属しているため、これらのトランザクションによる更新結果は１つのＷＡＬに記録される。ＷＡＬは以下のような情報で構成される。
・当該ＷＡＬを識別するためのＫｅｙ値
・トランザクションによる更新結果
【００３６】
ＷＡＬはＫＶデータであるため、（Ｋｅｙ,Ｖａｌｕｅ）の組で表される。Ｋｅｙ値としてデータグループで共通のＫｅｙ値にＷＡＬであることを示す接頭辞を付加したものを用いる。上記の例では、ＷＡＬ_ｉ１が当該ＷＡＬのＫｅｙ値となる。Ｖａｌｕｅ値にはトランザクションによる更新結果が含まれる。具体的には、以下のようにＷＡＬが構成される。
ＷＡＬ＝（Ｋｅｙ,Ｖａｌｕｅ）＝（ＷＡＬ_ｉ１,<[{ｉ１,ＡＡＡ,１０,５,１２０００},{ｂ１,ｉ１,ｕ３,１２０００},タイムスタンプ１],[{ｉ１,ＡＡＡ,１０,５,１３０００},{ｂ２,ｉ１,ｕ１,１３０００},タイムスタンプ２]>）
【００３７】
タイムスタンプは当該ＷＡＬに対するトランザクションが実行された順番を表している。タイムスタンプは単純な数値で表されていれば良い。このように、ＷＡＬはあるデータグループに対する更新履歴を表している。
【００３８】
ＷＡＬ特定のステップＳ００２でＷＡＬ管理手段１０１は、具体的には以下のようにＷＡＬを特定する。
【００３９】
まず、ＷＡＬ管理手段１０１は、入力されたトランザクションから、当該トランザクションで利用されるテーブルを抽出し、システムカタログ５００から当該トランザクションで利用されるテーブル間のデータグループに関係する情報を抽出する。
【００４０】
システムカタログ５００には、テーブルのスキーマ情報のほかに、どのテーブルがどの列でデータグループを構成するのかという情報が格納されている。例えば、図３に示すようなデータの場合には、「商品テーブルＴ００１と入札テーブルＴ００２でデータグループを構成し、データグループを構成する際のＫｅｙは商品ＩＤである。」という情報が格納されている。
【００４１】
ＷＡＬ管理手段１０１は、入力されたトランザクションとシステムカタログ５００の情報から、当該トランザクションが利用するデータグループを特定する。例えば、「入札テーブルＴ００２のＩＤがｂ１のレコードの入札額を１２０００に更新し、商品テーブルＴ００１の商品ＩＤがｉ１の最高金額を１２０００に更新する」というトランザクションが入力された場合は、当該トランザクションが利用するテーブルが商品テーブルＴ００１と入札テーブルＴ００２であることが分かり、システムカタログ５００から商品テーブルＴ００１と入札テーブルＴ００２が商品ＩＤでグループ化されていることが分かる。これにより、取得すべきＷＡＬがＷＡＬ_ｉ１であることが分かり、当該トランザクションが利用するＷＡＬが特定できる。
【００４２】

次に、ＷＡＬ特定のステップＳ００２で特定されたＷＡＬの情報を用いて、ＷＡＬ管理手段１０１がＷＡＬ記憶手段２００からＷＡＬを取得する（ステップＳ００３）。ここで、ＷＡＬ記憶手段２００にＷＡＬが存在する場合は、ＷＡＬ取得のステップＳ００４でＷＡＬ管理手段１０１がＷＡＬ記憶手段２００からＷＡＬを取得する。このとき、後の排他チェックで使用するために、当該ＷＡＬの最新のタイムスタンプをＷＡＬ記憶手段２００から取得しておく。

【００４３】

ここで、図５を用いてＷＡＬ記憶手段２００にＷＡＬがどのように保存されているのかを説明する。

【００４４】

ＷＡＬはＷＡＬ記憶手段２００の中でそれぞれのＷＡＬごとに履歴を持って保存されている。例えば、あるトランザクションがＷＡＬ_ｉ１を用いてトランザクションを実行すると、ＷＡＬ_ｉ１のタイムスタンプ１としてＷＡＬが蓄積され、次のトランザクションでＷＡＬ_ｉ１を用いてトランザクションを実行すると、ＷＡＬ_ｉ１のタイムスタンプ２としてＷＡＬが蓄積される。次のトランザクションでＷＡＬ_ｉ２を用いてトランザクションを実行するとＷＡＬ_ｉ２のタイムスタンプ１としてＷＡＬが蓄積される。

【００４５】

それぞれのＷＡＬに含まれるレコードについて、タイムスタンプの値が古いものから順番にＷＡＬのデータを確認すれば、各レコードの更新履歴や最新のレコード値を知ることができる。

【００４６】

ＷＡＬ記憶手段２００に該当するＷＡＬが存在しない場合は、レコード取得のステップＳ００６で、ＷＡＬ管理手段１０１がレコードデータ記憶手段４００から当該トランザクションで利用するレコードデータを取得する。

【００４７】

ここでレコードデータ記憶手段４００にレコードがどのように保存されているのかを説明する。レコードデータ記憶手段４００はＫＶＳであるため、レコードデータは（Ｋｅｙ,Ｖａｌｕｅ）の組で表されるデータとして保存されている。

【００４８】

図３に示すようなレコードデータは、主キーがＫｅｙ値となり、それ以外の列がＶａｌｕｅ値として保存されている。具体的には、以下のように保存されている。

（Ｋｅｙ,Ｖａｌｕｅ）＝（ｉ１,{ＡＡＡ,１０,５,１００００}）

（Ｋｅｙ,Ｖａｌｕｅ）＝（ｂ１,{ｉ１,ｕ３,９０００}）

【００４９】

ＷＡＬ記憶手段２００に該当するＷＡＬが存在する場合でも、ＷＡＬ記憶手段２００に保存されているＷＡＬの中に必要なレコードデータが全て含まれていない場合がある。以下にその場合の例を示す。

【００５０】

例えば、トランザクション１で以下のようなトランザクションを実行したとする。

「入札テーブルＴ００２のＩＤがｂ１のレコードの入札額を１２０００に更新し、商品テーブルＴ００１の商品ＩＤがｉ１の最高金額を１２０００に更新する」

このときＷＡＬは存在せず、新たにＷＡＬ_ｉ１のＷＡＬが作成され、ＷＡＬ記憶手段２００に保存されたとする。このとき、ＷＡＬ_ｉ１のＷＡＬとして保存されるＷＡＬは以下のようなものであり、レコードデータとしては、商品テーブルＴ００１の１行目のレコードと入札テーブルＴ００２の１行目のレコードが含まれる。

ＷＡＬ＝（ＷＡＬ_ｉ１,[{ｉ１,ＡＡＡ,１０,５,１２０００},{ｂ１,ｉ１,ｕ３,１２０００},１]）

次に、トランザクション２で以下のようなトランザクションを実行したとする。

「入札テーブルＴ００２のＩＤがｂ２のレコードの入札額を１３０００に更新し、商品テーブルＴ００１の商品ＩＤがｉ１の最高金額を１３０００に更新する。」

ＷＡＬ管理手段１０１は上記のトランザクション２が入力されると、ＷＡＬ_ｉ１を特定し、ＷＡＬ記憶手段２００からＷＡＬ_ｉ１のＷＡＬを取得する。このとき、ＷＡＬ_ｉ１にはタイムスタンプが１のＷＡＬのみが含まれており、当該ＷＡＬの中にはレコードデータとして、商品テーブルＴ００１の１行目のレコードと入札テーブルＴ００２の２行目のレコードが含まれている。しかし、トランザクション２では、商品テーブルＴ００１の１行目のレコードと、入札テーブルＴ００２の２行目のレコードを更新するため、ＷＡＬを取得しただけでは、トランザクションに必要なデータが全て揃わない。

【００５１】

このように、ＷＡＬ管理手段１０１はＷＡＬ記憶手段２００からＷＡＬを取得した後に、当該トランザクションの処理に必要なデータが当該ＷＡＬの中に全て含まれているかをチェックする（ステップＳ００５）。もし、必要なデータがＷＡＬの中に含まれていない場合は、レコードデータ記憶手段４００から必要な分のレコードを取得する（ステップＳ００６）。

【００５２】

ＷＡＬやレコードデータからトランザクションに必要なデータを全て取得したら、それらのデータを使用してトランザクションの処理を進める。そして、トランザクション実行のステップＳ００７で、ＷＡＬ管理手段１０１が、当該トランザクションによるデータグループの更新イメージを新たにＷＡＬとして作成する。この時点ではまだＷＡＬ管理手段１０１はＷＡＬ記憶手段２００に新たに作成したＷＡＬを書き込まない。

【００５３】

次に、参照・更新レコード抽出のステップＳ００８で、リード／ライトセット管理手段１０２は当該トランザクションで参照や更新したレコード情報を抽出する。具体的には、ＷＡＬ管理手段１０１から当該トランザクションで更新や参照したレコードの情報と、当該トランザクションのスタート時点のタイムスタンプ情報を取得する。なお、リード／ライトセット管理手段で保持しておくレコード情報はレコードデータ全体である必要はなく、各レコードのＫｅｙ値だけで十分である。

【００５４】

リード／ライトセット管理手段１０２は、排他チェック処理部１０３に対して、保持しているレコード情報を出力するが、このとき、参照したレコード情報と更新したレコード情報とを含む全てのレコード情報を出力してもよいし、更新したレコードの情報だけを出力してもよい。参照と更新のすべてのレコード情報を出力するか、更新したレコード情報だけを出力するかで、トランザクション間のＩｓｏｌａｔｉｏｎレベルを調整することができる。どの種別のレコード情報を出力するかは、システムが起動する前にシステム管理者などが予め設定しておけばよい。

【００５５】

次に、排他チェックのステップＳ００９で、排他チェック処理部１０３は当該トランザクションがスタートしてから現時点までにＷＡＬ記憶手段２００に追加されたＷＡＬデータと、リード／ライトセット管理手段１０２から渡される当該トランザクションで参照や更新したレコードとを比較して（ステップＳ０１０）、同じレコードが存在する場合はトランザクションを失敗させ（ステップＳ０１２）、同じレコードが存在しない場合はトランザクションを成功させる（ステップＳ０１１）。

【００５６】

具体的には、排他チェック処理部１０３はまず、リード／ライトセット管理手段１０２から当該トランザクションで参照や更新されたレコードデータとスタート時点のタイムスタンプの値を受け取る。次に、ＷＡＬ記憶手段２００から当該トランザクションで使用されたＷＡＬを取得する。

【００５７】

ＷＡＬ記憶手段２００から取得したＷＡＬの最新のタイムスタンプがリード／ライトセット管理手段１０２から受け取ったスタート時点のタイムスタンプと同じ値の場合は、当該トランザクションがスタートしてから現時点までの間に当該ＷＡＬに関連する他のトラ

ンザクションが実行されていないことになるため、トランザクションを成功させる。
【００５８】
ＷＡＬ記憶手段２００から取得したＷＡＬの最新のタイムスタンプが、リード／ライトセット管理手段１０２から受け取ったスタート時点のタイムスタンプの値より大きな値の場合は、当該トランザクションがスタートしてから現時点までの間に当該ＷＡＬに関連する他のトランザクションが実行されていることを意味するため、排他違反がないかチェックする。
【００５９】
排他違反チェックは以下のようにして実行される。まず、ＷＡＬデータ分解手段１１３が、ＷＡＬ記憶手段２００から取得したＷＡＬデータのうち、リード／ライトセット管理手段１０２から受け取ったタイムスタンプの値より大きなタイムスタンプの値を持つＷＡＬデータを全て抽出する。そして、抽出したＷＡＬデータをレコード単位に分解して当該ＷＡＬデータに含まれるレコードデータを抽出する。
【００６０】
次に、レコードデータ照合手段１２３が、抽出したＷＡＬデータに含まれるレコードデータと、リード／ライトセット管理手段１０２から受け取ったレコードデータに一致するものが含まれるかどうかをチェックする。もし、一致するものが１つでも存在する場合は排他違反が発生しているため、トランザクションを失敗させる。一致するものが存在しない場合は排他違反が発生していないため、トランザクションを成功させる。
【００６１】
排他チェック処理部１０３がトランザクションを成功させる場合、失敗させる場合を説明する概念図を図６、図７、図８に示す。
【００６２】
図６は、トランザクションがスタートしてから排他チェック処理部１０３が排他チェックを実行するまでの間に当該ＷＡＬに更新が無かった場合の例である。この場合、当該トランザクションが実行されている間に当該ＷＡＬに関連する他のトランザクションが実行されていないため、当該トランザクションは成功する。
【００６３】
図７は、トランザクションがスタートしてから排他チェック処理部１０３が排他チェックを実行するまでの間に当該ＷＡＬに更新があった場合の例である。この場合、排他チェック処理部１０３で持っているスタート時点のタイムスタンプ値より大きなタイムスタンプ値を持つＷＡＬデータを抽出する。図７の例では、タイムスタンプが３のＷＡＬデータを取得する。図７の例では、タイムスタンプが３のＷＡＬデータにはレコードＡとレコードＤが含まれている。つまり、当該トランザクションがスタートしてから排他チェック処理部１０３が排他チェックを実行するまでの間に、他のトランザクションがレコードＡとレコードＤを更新するトランザクションを実行したことを意味している。ここで、当該トランザクションを実行するとレコードＡに関して排他違反が発生するため、当該トランザクションを失敗させる。
【００６４】
図８は、トランザクションがスタートしてから排他チェック処理部１０３が排他チェックを実行するまでの間に当該ＷＡＬに更新があった場合の例である。この場合、排他チェック処理部１０３で持っているスタート時点のタイムスタンプ値以上のタイムスタンプ値を持つＷＡＬデータを抽出する。図８の例ではタイムスタンプ値が３のＷＡＬデータを取得する。図８の例では、タイムスタンプが３のＷＡＬデータにはレコードＣとレコードＤが含まれている。つまり、当該トランザクションがスタートしてから排他チェック処理部１０３が排他チェックを実行するまでの間に、他のトランザクションがレコードＣとレコードＤを更新するトランザクションを実行したことを意味している。ここで、当該トランザクションを実行してもレコードレベルでは排他違反は発生していないため、当該トランザクションを成功させる。
【００６５】

排他チェック処理部１０３で排他違反が検出されなかった場合は、排他チェック処理部１０３はＷＡＬ管理手段１０１にトランザクションのコミットを要求する。ＷＡＬ管理手段１０１はトランザクションのコミットが要求されると、当該トランザクションのＷＡＬデータをＷＡＬ記憶手段２００に書き込む（ステップＳ０１１）。このとき、書き込むＷＡＬのタイムスタンプの値は、ＷＡＬ記憶手段２００に記憶されている最新のタイムスタンプの次の値とする。例えば図６の場合、更新後のレコードＡ、Ｂに値３のタイムスタンプを付加して書き込む。また例えば図８の場合、更新後のレコードＡ、Ｂに値４のタイムスタンプを付加して書き込む。

【００６６】

排他チェック処理部１０３で排他違反が検出された場合は、排他チェック処理部１０３はＷＡＬ管理手段１０１にトランザクションのアボートを要求する。ＷＡＬ管理手段１０１はトランザクションのアボートが要求されると、当該トランザクションのＷＡＬデータを破棄する（ステップＳ０１２）。

【００６７】

次に、本実施形態においてＷＡＬデータをレコードデータ記憶手段４００に反映する動作について説明する。ＷＡＬデータの反映は図９に示すフローチャートに従って動作を行う。

【００６８】

まず、ＷＡＬデータ取得のステップＳ１０１でＷＡＬ反映手段３００がＷＡＬ記憶手段２００から、記憶されているＷＡＬデータを全て取得する。

【００６９】

ＷＡＬ記憶手段２００にＷＡＬデータが存在しない場合はＷＡＬ反映の処理を終了する（ステップＳ１０２のｎｏ）。

【００７０】

ＷＡＬ記憶手段２００にＷＡＬデータが存在する場合（ステップＳ１０２のｙｅｓ）は、ＷＡＬデータ反映のステップＳ１０３でＷＡＬ反映手段３００がＷＡＬ記憶手段２００から取得したＷＡＬデータに含まれるレコードデータを抽出し、抽出したレコードデータをレコードデータ記憶手段４００に反映させる。

【００７１】

ＷＡＬ反映手段３００がＷＡＬ記憶手段２００から取得したＷＡＬデータには同じレコードデータが複数含まれている場合がある。同じレコードデータが複数含まれている場合は、タイムスタンプがもっとも大きな値のＷＡＬに含まれるレコードデータが当該レコードデータについて最新のデータであるため、これを反映する。

【００７２】

図１０を用いて具体的に説明する。ＷＡＬ反映手段３００がＷＡＬ記憶手段２００から図１０に示すようなＷＡＬを取得した場合、レコードＡについては、タイムスタンプが１のＷＡＬデータに含まれるものが最新のため、このデータをレコードデータ記憶手段４００に反映する。レコードＢとレコード Ｃについては、タイムスタンプが３のＷＡＬデータに含まれるものが最新のため、このデータをレコードデータ記憶手段４００に反映する。

【００７３】

最後に、ＷＡＬデータ削除のステップＳ１０４でＷＡＬ反映手段３００がＷＡＬ記憶手段２００から抽出し、反映したＷＡＬデータを削除する。ＷＡＬ反映手段３００がＷＡＬ記憶手段２００からデータを取得し、反映するまでの間にトランザクションが実行され、新たなＷＡＬが追加されている可能性があるが、新たに追加されたＷＡＬについては削除しない。

【００７４】

あるレコードについて、ＷＡＬにデータが存在する場合は、ＷＡＬに保存されているレコードデータが当該レコードについて最新の情報であり、ＷＡＬ反映手段３００がレコードデータ記憶手段４００にレコードデータを反映した時点で当該レコードに対する最新の

データがレコードデータ記憶手段４００に保存される。あるレコードについてＷＡＬにデータが存在しない場合は、レコードデータ記憶手段４００に保存されているレコードデータが当該レコードについて最新の情報となる。

【００７５】

なお、アプリケーションから入力されたトランザクションを処理するプロセスと、ＷＡＬを反映するプロセスはそれぞれ並列に動作している。ＷＡＬを反映するプロセスはタイマーによって一定時間ごとに行われてもよいし、システム管理者等が任意のタイミングで行っても良い。

【００７６】

[効果]

本実施形態は、ＫＶＳに対して複数レコードのトランザクションをサポートするシステムに対して、トランザクションの排他制御を行う際に、トランザクションを実行するデータグループ単位で排他制御を行うのではなく、当該トランザクションで参照や更新したレコード単位で排他制御を行うことで、トランザクションの排他単位を小さくすることができ、トランザクションの同時実効性を高めスループットを上げることができる。

【００７７】

[第２の実施形態]

次に、本発明の第２の実施形態について説明する。

【００７８】

図１１に示すように、本実施形態は図１に示した第１の実施形態の構成要素に加えて、ＷＡＬ反映手段３００の中にＷＡＬ削除制御手段３０１を備える点で異なる。

【００７９】

本実施形態では、ＷＡＬ反映手段３００はＷＡＬデータをレコードデータ記憶手段４００に反映させた後に、反映させた全てのＷＡＬデータを削除するのではなく、一定量のＷＡＬデータを削除せず残しておく。

【００８０】

図１２に本実施形態におけるＷＡＬ反映手段３００によるＷＡＬデータ管理の概念図を示す。ＷＡＬ反映手段３００が処理実行時にタイムスタンプ１からタイムスタンプ６までのＷＡＬデータが存在したとする。ＷＡＬ削除制御手段３０１には、ＷＡＬデータ反映後のＷＡＬデータ削除時に、反映済みのＷＡＬデータのうち、どれぐらいのＷＡＬデータを削除せずに残しておくかの情報を持っている。図１２に示す例ではＷＡＬ削除制御手段３０１は３個のＷＡＬデータを削除せずに残しておくという情報を持っている。なお、タイムスタンプ７からタイムスタンプ９までのＷＡＬデータは、ＷＡＬ反映手段３００によってＷＡＬ反映処理が開始してからＷＡＬ反映処理が終了するまでの間に実行されたトランザクションによって新たに生成されたＷＡＬデータである。このように新たなＷＡＬデータはトランザクションの実行によって逐次蓄積されている。

【００８１】

本実施形態におけるＷＡＬ反映処理は基本的には図９に示すフローチャートと同様の手順で動作を行う。

【００８２】

まず、ＷＡＬデータ取得のステップＳ１０１でＷＡＬ反映手段３００がＷＡＬ記憶手段２００からそこに記憶されているＷＡＬデータを全て取得する。

【００８３】

次に、ＷＡＬデータ反映のステップＳ１０２でＷＡＬ反映手段３００がＷＡＬ記憶手段２００から取得したＷＡＬデータに含まれるレコードデータを抽出し、抽出したレコードデータをレコードデータ記憶手段４００に反映させる。

【００８４】

最後に、ＷＡＬデータ削除のステップＳ１０３で、ＷＡＬ削除制御手段３０１が、反映したＷＡＬデータのうちタイムスタンプが大きな値のものから順番にＮ個（Ｎは自然数）だけ残して、ＷＡＬデータを削除する。すなわち、タイムスタンプの新しい上位所定個数

のＷＡＬデータは削除せずに残しておく。
【００８５】
次に、反映したＷＡＬデータを全て削除せずに一部残しておくことの効果について説明する。トランザクションの実行プロセスとＷＡＬ反映のプロセスはそれぞれ並列に動作しているため、トランザクションがスタートしてからコミットするまでの間にＷＡＬ反映のプロセスが動作する可能性がある。
【００８６】
もし、あるトランザクションがスタートしてからコミットするまでの間にＷＡＬ反映処理が動作した場合、当該トランザクションに関連するＷＡＬデータはＷＡＬ記憶手段２００から全て削除されてしまう。ＷＡＬデータが削除された後に当該トランザクションの排他チェックを実行しようとしても、ＷＡＬデータが存在しないため排他違反があるのかないのかが判断できない。このため、本来成功するはずのトランザクションも失敗することになる。これに対して、ＷＡＬ反映処理によって全てのデータが削除されず、トランザクションの排他チェックのために十分なＷＡＬデータが残っていれば当該トランザクションの排他違反を正確に検出できる。
【００８７】
例えば、本発明の第１の実施形態で参照した図６乃至図８においては、右側の破線内のＷＡＬの状態に基づいて、図６および図８の場合は排他違反無し、図７の場合は排他違反有りが検出された。若し、図６乃至図８において、右側の破線内のＷＡＬが全て削除されてしまっていると、そのような検出は困難になる。従って、安全を考えて図６乃至図８の全てのケースにおいて排他違反有りと判断せざるを得ず、トランザクションの同時実行性が低下する。
【００８８】
このように、本実施形態では、ＷＡＬ反映手段３００によって反映したＷＡＬデータを全て削除するのではなく、一部のＷＡＬデータを残しておくことによって、ＷＡＬデータが存在しないためにトランザクションの実行が失敗してしまうという事態を避け、本来失敗しないはずのトランザクションを成功させることができ、トランザクションの同時実行性を向上させスループットを上げることができる。
【００８９】
なお、どれぐらいの量のＷＡＬを削除せずに残しておくかは、求めるトランザクションの同時実行性、使用する記憶容量とのトレードオフである。使用する記憶容量を抑えるためには、残しておくＷＡＬは少なくすべきであり、トランザクションの同時実行性を上げるためには、残しておくＷＡＬは多くすべきである。
【００９０】
ＷＡＬ削除制御手段３０１が管理する、残しておくＷＡＬの量は固定値であってもよいし、ＷＡＬが存在しないために排他チェックが行えず失敗したトランザクションの量からフィードバックを受けて自動で調整されてもよい。つまり、排他チェックが行えず失敗したトランザクションの量が多いならば残しておくＷＡＬの量を大きくし、排他チェックが行えずに失敗したトランザクションの量が少ないなら残しておくＷＡＬの量を少なくすればよい。
【００９１】
また、本実施形態におけるＷＡＬの構造は、以下のようになっている。
・当該ＷＡＬを識別するためのＫｅｙ値
・データグループに対する更新結果
・ＷＡＬ反映位置
【００９２】
即ち、第１の実施形態と比較して、新たにＷＡＬ反映位置が追加されている。これは、ＷＡＬデータのうちレコードデータ記憶手段４００に反映済みのデータ位置を示している。図１２に示すＷＡＬデータの例では、ＷＡＬ反映位置はタイムスタンプ６となる。ＷＡＬ管理手段１０１などがＷＡＬ記憶手段２００からＷＡＬデータを取得した際にＷＡＬ反

映位置を参照することによって、ＷＡＬデータのうち、どのデータがレコードデータ記憶手段４００に反映済みであるのかが分かる。同様に、ＷＡＬ反映手段３００は、ＷＡＬ記憶手段２００からＷＡＬデータを取得する際にＷＡＬ反映位置を参照することによって、ＷＡＬデータのうち、どのデータがレコードデータ記憶手段４００に既に反映済みであるのかが分かる。

【００９３】

[第３の実施形態]

次に、本発明の第３の実施形態について説明する。

【００９４】

図１３に示すように、本実施形態は図１に示した第１の実施形態の構成要素に加えて、ＷＡＬ反映タイミング制御手段３１０を備える点で異なる。

【００９５】

本実施形態では、ＷＡＬ反映手段３００は一定時間ごとやシステム管理者等による動作指示ではなく、ＷＡＬ反映タイミング制御手段３１０からの指示によってＷＡＬ反映処理を実行する。

【００９６】

ＷＡＬ反映処理はすべてのＷＡＬに含まれるレコードデータをレコードデータ記憶手段４００に反映するため、負荷が高い処理である。このため、アプリケーションから頻繁にトランザクションが入力されているようなシステムに高い負荷がかかっているときにＷＡＬ反映処理が動作するとシステムにさらに負荷がかかってしまい、トランザクションの処理性能に悪影響を及ぼす。他方、ＷＡＬはレコードデータに対する更新履歴が全て含まれているため、ＷＡＬを反映せずに残しておくと、ＷＡＬを保存するための記憶装置の使用容量が大きくなってしまう。

【００９７】

ＷＡＬ反映タイミング制御手段３１０は、現在のシステムの負荷状況や記憶装置の使用量を監視して、システムの負荷が低い場合や、記憶装置の使用量が大きくなってきた場合にＷＡＬ反映手段３００に動作を指示する。

【００９８】

例えば、以下のような評価式を用いて、評価値がある閾値を超えた場合にＷＡＬ反映タイミング制御手段３１０は、ＷＡＬ反映手段３００に動作を指示する。

評価値＝Ｗ１＊ＣＰＵのアイドル状況＋Ｗ２＊（記憶装置の使用量/記憶装置の全容量）

【００９９】

上記評価式において、Ｗ１やＷ２は重み係数であり、システムの負荷の状況と記憶装置の使用量のどちらを優先するかを調整することができる。例えば、Ｗ１を大きくした場合は、よりシステムの負荷状況を優先した動作になる。Ｗ１やＷ２などの重み係数や、評価式に対する閾値などはシステム管理者がシステム起動前に予め設定しておけばよい。

【０１００】

本実施形態におけるＷＡＬ反映処理のフローチャートを図１４に示す。

【０１０１】

まず、評価値計算のステップＳ３０１で、ＷＡＬ反映タイミング制御手段３１０が評価値の計算を行う。

【０１０２】

次に、ＷＡＬ反映タイミング制御手段３１０は、ステップＳ３０１で計算した評価値が閾値を超えているかどうかを判定する（ステップＳ３０２）。評価値が閾値を超えていない場合は、評価値計算のステップＳ３０１に戻る。評価値が閾値を超えている場合は、ＷＡＬ反映タイミング制御手段３１０はＷＡＬ反映手段３００に動作指示を行う。

【０１０３】

ＷＡＬ反映タイミング制御手段３１０から動作指示を受けたＷＡＬ反映手段３００は、ＷＡＬ反映処理を実行する。ステップＳ３０４以降の動作は図９に示したＷＡＬ反映の動作フローと同じため説明を割愛する。

【０１０４】
このように、本実施形態では、ＷＡＬ反映タイミング制御手段３１０がシステムの負荷状況や記憶装置の使用量などを考慮してＷＡＬ反映の動作タイミングを制御することで、本システムが稼動しているコンピュータの資源を有効に使うことができる。
【０１０５】
[第４の実施形態]
図１５は、本発明の第４の実施形態のブロック図である。図１５に示すように、本実施形態は、１以上のトランザクション処理部１と、レコードデータ記憶部２と、ＷＡＬ記憶部３と、ＷＡＬ反映部４とを有する。
【０１０６】
レコードデータ記憶部２は、半導体メモリや磁気ディスク等で構成され、キーとバリューの組で表現されるレコードを記憶する。
【０１０７】
ＷＡＬ記憶部３は、半導体メモリや磁気ディスク等で構成され、複数の上記レコードを１つのキーバリューデータに集約したデータグループ単位で、その更新履歴を記憶する。好ましくは、ＷＡＬ記憶部３は、キーとバリューの組で表現される更新ログを記憶する。上記キーは、複数の上記レコードを集約したデータであるデータグループを一意に識別する値である。また上記バリューは、上記データグループの更新履歴を表す。さらに上記更新履歴は、１以上の部分更新履歴を有する。そして、１つの上記部分更新履歴は、上記データグループを構成する上記レコードの更新内容と更新の順番を表すタイムスタンプとを有する。
【０１０８】
トランザクション処理部１は、図示しないアプリケーションプログラム等から入力されたトランザクションを実行する機能を有する。トランザクション処理部１は、上記データグループを参照および更新するトランザクションのコミット時、ＷＡＬ記憶部３に記憶された更新履歴に基づいて、レコード単位で排他違反の有無を判定し、排他違反がなければ当該トランザクションを成功させ、その更新履歴をＷＡＬ記憶部３に記録する。好ましくは、トランザクション処理部１は、主な機能部として、ＷＡＬ管理部５と排他チェック処理部６とを有する。
【０１０９】
ＷＡＬ管理部５は、入力されたトランザクションで使用する上記データグループを構成する上記レコードの最新の値をＷＡＬ記憶部３およびレコードデータ記憶部２から取得する機能を有する。またＷＡＬ管理部５は、当該データグループに対応する上記更新ログ中の最新の上記タイムスタンプをスタート時タイムスタンプとしてＷＡＬ記憶部３から取得する機能を有する。
【０１１０】
排他チェック処理部６は、トランザクション処理部１で実行しているトランザクションのコミット時、当該トランザクションのスタート時に取得しておいた上記スタート時タイムスタンプと、ＷＡＬ記憶部３に現に記憶されている上記データグループの最新のタイムスタンプであるコミット時タイムスタンプとを比較する機能を有する。また排他チェック処理部６は、上記の比較の結果、スタート時タイムスタンプとコミット時タイムスタンプとが相違するときは、スタート時タイムスタンプより後のタイムスタンプを有する部分更新履歴中に、当該トランザクションが使用したレコードと一致するレコードが存在するか否かを判定する機能を有する。さらに排他チェック処理部６は、上記比較と上記判定の結果とに基づいて、上記トランザクションを成功させるか、失敗させるかを決定し、成功させると決定した場合、上記トランザクションで更新したレコードの更新内容とコミット時タイムスタンプより後のタイムスタンプとを有する新たな部分更新履歴を、上記データグループに対応してＷＡＬ記憶部３に現に記憶されている更新ログに追加する機能を有する。
【０１１１】

ＷＡＬ反映部４は、ＷＡＬ記憶部３に記憶されている更新履歴中のレコードの更新内容をレコードデータ記憶部２に反映する機能を有する。

【０１１２】

本実施形態のトランザクションシステムは、例えば、ＭＰＵ等のプロセッサとそれに接続されたメモリ、入出力装置、通信インターフェイス等を有するコンピュータと、当該コンピュータで実行されるプログラムとで実現することができる。プログラムは、磁気ディスク装置等のコンピュータ可読記録媒体に記録されており、コンピュータの立ち上げ時にＭＰＵ等のプロセッサに読み取られ、そのプロセッサの動作を制御することにより、そのプロセッサ上にＷＡＬ管理部５、排他チェック処理部６、ＷＡＬ反映部４を実現する。

【０１１３】

次に本実施形態の動作を説明する。

【０１１４】

トランザクション処理部１は、入力されたトランザクションを実行する。そして、トランザクション処理部１は、データグループを参照および更新するトランザクションのコミット時、ＷＡＬ記憶部３に記憶された更新履歴に基づいて、レコード単位で排他違反の有無を判定し、排他違反がなければ当該トランザクションを成功させ、その更新履歴をＷＡＬ記憶部３に記録する。その後、ＷＡＬ反映部４は、ＷＡＬ記憶部３に記憶されている更新履歴中のレコードの更新内容をレコードデータ記憶部２に反映する。

【０１１５】

より好ましくは、本実施形態は以下のように動作する。

【０１１６】

トランザクション処理部１は、トランザクションが入力されると、そのトランザクションで使用するデータグループを構成するレコードの最新の値をＷＡＬ記憶部３およびレコードデータ記憶部２から取得し、また、当該データグループに対応する更新ログ中の最新のタイムスタンプをスタート時タイムスタンプとしてＷＡＬ記憶部３から取得し、トランザクションの実行を開始する。

【０１１７】

トランザクション処理部１は、実行しているトランザクションのコミット時、そのトランザクションのスタート時に取得しておいた上記スタート時タイムスタンプと、ＷＡＬ記憶部３に現に記憶されているデータグループの最新のタイムスタンプであるコミット時タイムスタンプとを比較する。次にトランザクション処理部１は、スタート時タイムスタンプとコミット時タイムスタンプとが相違するときは、スタート時タイムスタンプより後のタイムスタンプを有する全ての部分更新履歴をＷＡＬ記憶部３から読み出し、それらの部分更新履歴中に、当該トランザクションが使用したレコード（参照または更新したレコード、或いは更新したレコード）と一致するレコードが存在するか否かを判定する。

【０１１８】

トランザクション処理部１は、上記の比較と判定の結果とに基づいて、上記トランザクションを成功させるか、失敗させるかを決定する。具体的には、スタート時タイムスタンプとコミット時タイムスタンプとが同じであるか、同じでなくても、スタート時タイムスタンプより後のタイムスタンプを有する全ての部分更新履歴中に、当該トランザクションが使用したレコード（参照または更新したレコード、或いは更新したレコード）と一致するレコードが存在しなければ、トランザクションを成功させる。しかし、スタート時タイムスタンプとコミット時タイムスタンプとが異なっており、しかも、スタート時タイムスタンプより後のタイムスタンプを有する部分更新履歴中に、当該トランザクションが使用したレコード（参照または更新したレコード、或いは更新したレコード）と一致するレコードが存在していれば、トランザクションを失敗させる。

【０１１９】

トランザクション処理部１は、トランザクションを成功させると決定した場合、当該トランザクションで更新したレコードの更新内容とコミット時タイムスタンプより後のタイムスタンプとを有する新たな部分更新履歴を作成して、当該トランザクションが使用した

データグループに対応してＷＡＬ記憶部３に現に記憶されている更新ログに追加する。更新ログは、キーとバリューの組で表現される１つのキーバリューデータであるため、ＣＡＳコマンド等を使用することで、データ操作のアトミック性を保証することができる。

【０１２０】

他方、ＷＡＬ反映部４は、例えば定期的に、ＷＡＬ記憶部３に記憶されている更新ログ中のレコードの更新内容をレコードデータ記憶部２に反映する。レコードは、キーとバリューの組で表現される１つのキーバリューデータであるため、ＣＡＳコマンド等を使用することで、データ操作のアトミック性を保証することができる。

【０１２１】

こうして本実施形態によれば、複数のレコードを１つのＫＶデータにまとめてトランザクションを実行するシステムにおけるトランザクションの同時実行性を高めることが可能である。その理由は、レコード単位で楽観的な排他を行うためである。

【０１２２】

以上、上記各実施形態を参照して本発明を説明したが、本発明は、上述した実施形態に限定されるものではない。本発明の構成や詳細には、本発明の範囲内で当業者が理解しうる様々な変更をすることができる。

【０１２３】

なお、本発明は、日本国にて２０１２年１０月２９日に特許出願された特願２０１２－２３７５９７の特許出願に基づく優先権主張の利益を享受するものであり、当該特許出願に記載された内容は、全て本明細書に含まれるものとする。

【産業上の利用可能性】

【０１２４】

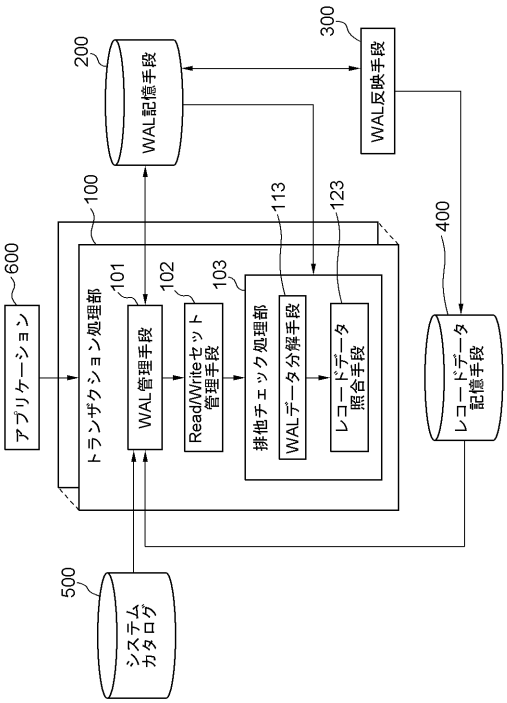
本発明は、データ永続化手段としてＫＶＳを用いてトランザクションを実行するシステムにおいて利用可能である。

【符号の説明】

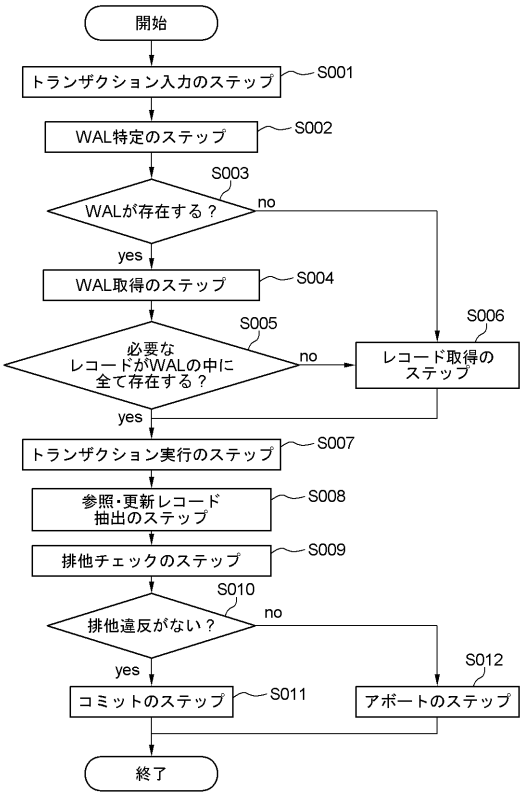
【０１２５】

１…トランザクション処理部
２…レコードデータ記憶部
３…ＷＡＬ記憶部
４…ＷＡＬ反映部
５…ＷＡＬ管理部
６…排他チェック処理部

【図１】



【図２】



【図３】

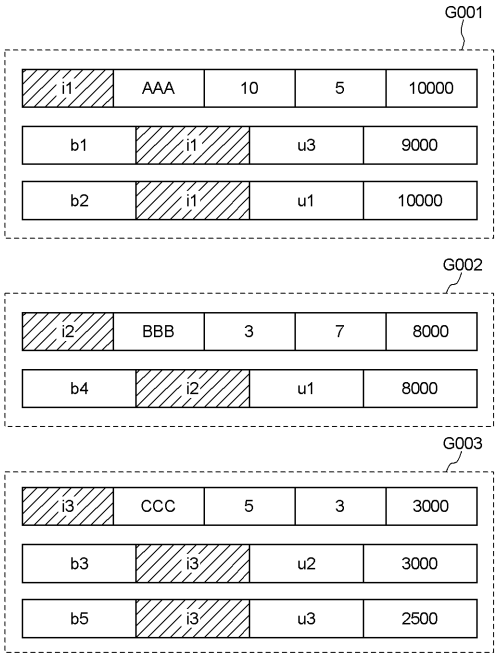
商品テーブル　T001

商品ID	商品名	在庫	入札数	最高金額
i1	AAA	10	5	10000
i2	BBB	3	7	8000
i3	CCC	5	3	3000

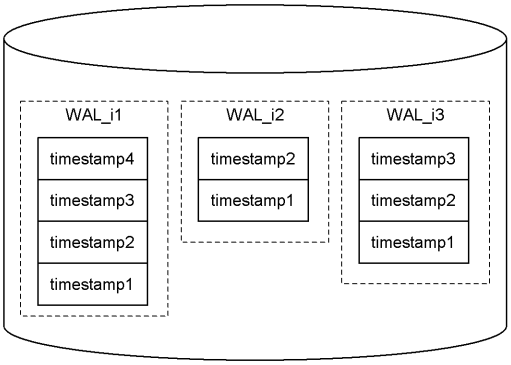
入札テーブル　T002

ID	商品ID	ユーザID	入札金額
b1	i1	u3	9000
b2	i1	u1	10000
b3	i3	u2	3000
b4	i2	u1	8000
b5	i3	u3	2500

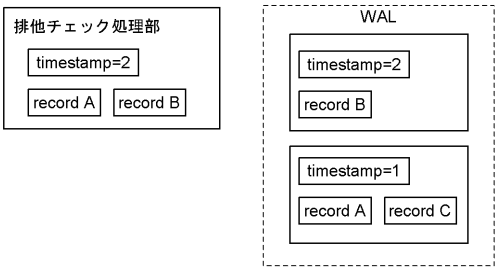
【図４】



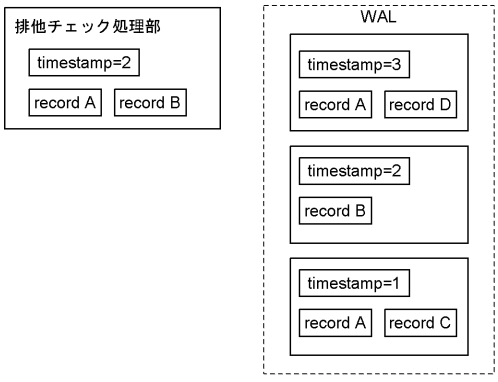
【図５】


WAL_i1
timestamp4
timestamp3
timestamp2
timestamp1
WAL_i2
timestamp2
timestamp1
WAL_i3
timestamp3
timestamp2
timestamp1

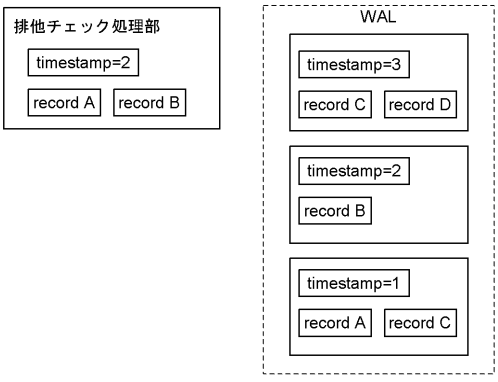
【図６】


排他チェック処理部
timestamp=2
record A
record B
WAL
timestamp=2
record B
timestamp=1
record A
record C

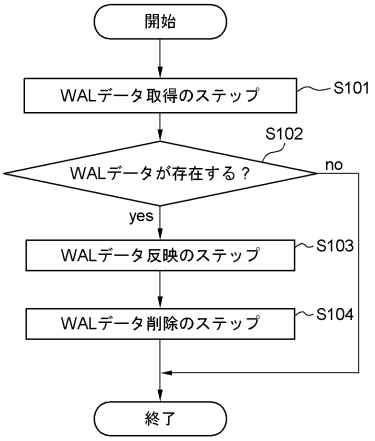
【図７】


排他チェック処理部
timestamp=2
record A
record B
WAL
timestamp=3
record A
record D
timestamp=2
record B
timestamp=1
record A
record C

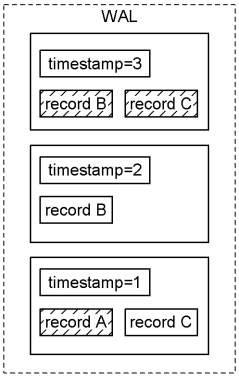
【図８】


排他チェック処理部
timestamp=2
record A
record B
WAL
timestamp=3
record C
record D
timestamp=2
record B
timestamp=1
record A
record C

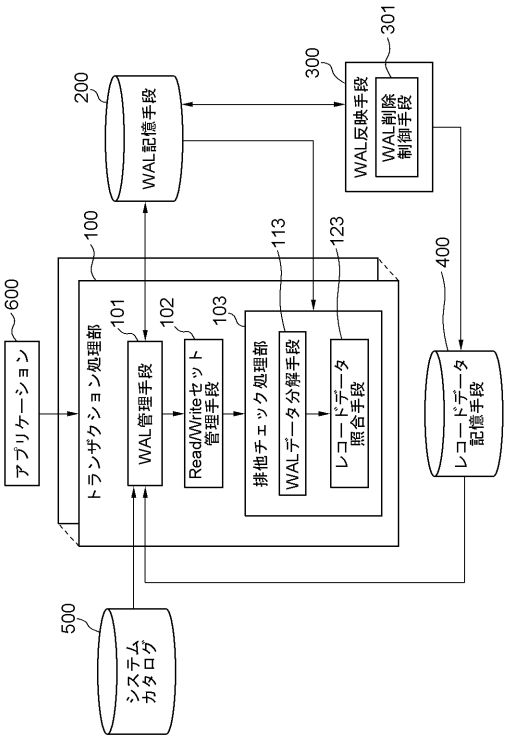
【図９】



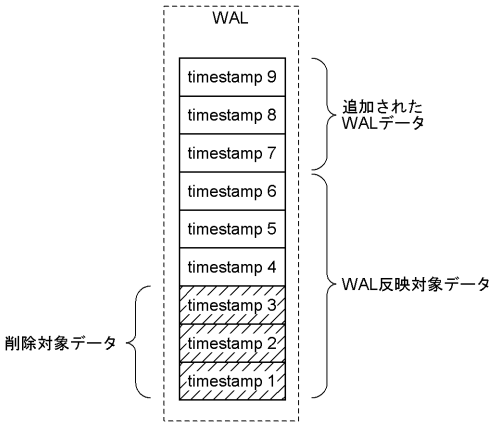
【図１０】



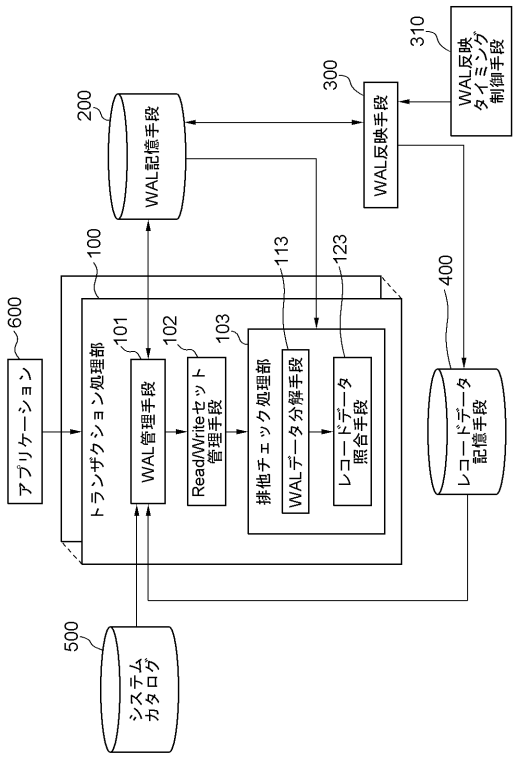
【図１１】



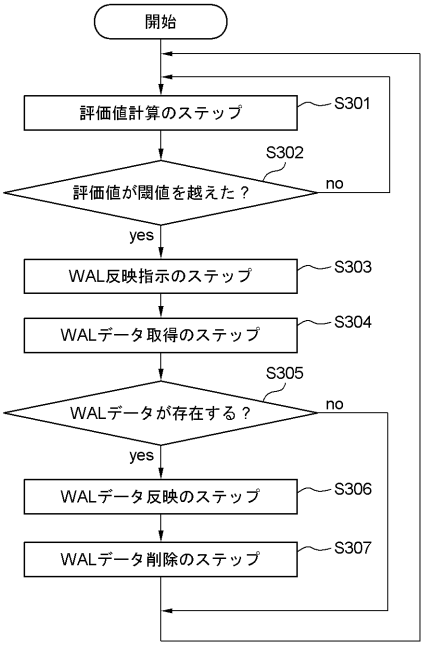
【図１２】



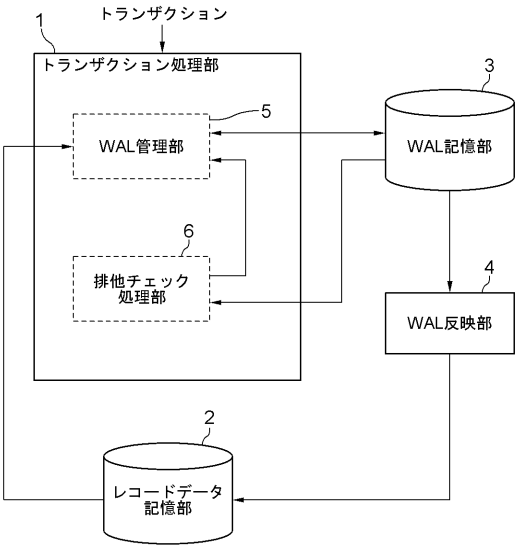
【図１３】


310
WAL反映タイミング制御手段
300
WAL反映手段
200
WAL記憶手段
100
113
123
400
600
101
102
103
アプリケーション
トランザクション処理部
WAL管理手段
Read/Writeセット管理手段
排他チェック処理部
WALデータ分解手段
レコードデータ照合手段
レコードデータ記憶手段
500
システムカタログ

【図１４】


開始
評価値計算のステップ
S301
S302
評価値が閾値を越えた？
no
yes
WAL反映指示のステップ
S303
WALデータ取得のステップ
S304
S305
WALデータが存在する？
no
yes
WALデータ反映のステップ
S306
WALデータ削除のステップ
S307

【図１５】


1
トランザクション
トランザクション処理部
5
WAL管理部
6
排他チェック処理部
3
WAL記憶部
4
WAL反映部
2
レコードデータ記憶部

フロントページの続き

(56)参考文献 国際公開第２００７／０６３９４５（WO，Ａ１）
特開２００９－２６０２９（ＪＰ，Ａ）
特開２００１－１０１０４５（ＪＰ，Ａ）
祐成光樹 他，"ビッグデータの活用に最適なスケールアウト型 新データベース「InfoFrame Relational Store」"，NEC技報，日本電気株式会社，２０１２年 9月 1日，Vol.65，No.2，p.30-33
Junichi Tatemura et al.，"Microsharding: a declarative approach to support elastic OLTP workloads"，ACM SIGOPS Operating Systems Review，２０１２年 1月，Vol.46，No.1，p.4-11

(58)調査した分野(Int.Cl.，ＤＢ名)
Ｇ０６Ｆ９／４６
１２／００
１７／３０
ＪＳＴＰｌｕｓ（ＪＤｒｅａｍＩＩＩ）