オンライントランザクション処理中におけるデータベース整合性チェック方式

6G－5

小林敦司　小林伸幸
NTT情報通信研究所

１．はじめに

データベース(DB)の適用分野の拡大に伴い、24時間無中断でDBサービスを運用する必要性が高まってきている。一般にDB破壊を発見するためには、トランザクションを止めた静的なDBの状態で整合性チェックを行うが、24時間無中断運用を前提とした場合、DB内容が刻々と変化する動的なDB状態での整合性チェックを実現する必要がある。

本稿では、オンライントランザクションを止めることなく、動的DB状態でDB全体の整合性をチェックする方式を提案するとともに、その実現方法について述べる。

２．DB整合性チェックの内容

DB整合性チェックとはDB破壊の早期発見のために、DBの格納構造をチェックすることを指す。DBの整合性チェック内容を以下に示す。

- フォーマットチェック
  単一のフィールドの設定値の正当性をチェック（ディレクトリ／スキーマ、テーブル、インデックスの各種フィールド）
- 相関チェック
  複数のフィールド間の対応関係が正しく設定されているかのチェック（ディレクトリースキーマ間、インデックスーレコード間等）

３．動的DB状態のDB整合性チェック方式

従来の整合性チェックはトランザクションを止め、DB内容が変化しない静的なDBの状態で行うのが一般的であった。しかし24時間無中断運用を前提とした場合、DB内容が刻々と変化する動的DB状態になっており、一時的にDB整合性が保たれていない状況が発生する（例：インデックス更新中はレコードとインデックスの対応関係が矛盾状態となる）。そのため、従来のようにトランザクションを止めてDB整合性チェックを行うことができない。そこで一時的なDB矛盾状態を考慮したDB整合性チェック方式として以下の案が考えられる。

案1:一時的な矛盾状態のままチェックする。矛盾を検出した場合は数回リトライし一定回数以上繰り返しても矛盾状態のままだと矛盾と判定する。

案2:DBの整合がとれた局所的なブロックを作り、その局所ブロックの整合性チェックを積み上げることで、DB全体の整合性をチェックする。

案1はチェック対象間のDB内容が保証されない状態でチェックを進めるので、チェック対象レコード／インデックス参照中や、チェック対象レコード／インデックスからチェック先のレコード／インデックスを参照する間に、そのチェック対象あるいはチェック先が更新／削除される可能性がある。そのために、チェックを進めて破壊（矛盾状態）と判定されても、それが更新中によるものか本当の破壊なのかは判定できない。リトライを行っても依然として双方のレコードの対応がとれていない場合も有り得る。

そこで動的DB状態でチェックの完全性を得るために案2を採用する。案2はチェック対象となるフィールドについて整合性がとれた状態を作るために局所的に参照ロックをかけ、DB状態をチェックし、すぐにロックを解除する。これを繰り返すことでDB全体の整合性をチェックする方法である（局所ブロック積み上げ方式）。ロック箇所はフォーマットチェックでは単一のフィールドとなり、相関チェックでは複数のフィールドとなる。

４．局所ブロック積み上げ方式によるチェック処理方式

本稿ではインデックスーレコード相関チェックを例にとって方式を述べる。インデックスーレコード相関チェックは以下のようなチェックが必要となる。

①レコードに対応するインデックスの存在チェック
②インデックスに対応するレコードの存在チェック

A Check Method of Database Integrity in On-line Transaction Processing
Atsushi Kobayashi and Nobuyuki Kobayashi
NTT Information and Communication Systems Laboratories
1-2356 Take, Yokosuka, Kanagawa 238-03, Japan

① レコード→インデックス相関チェック

1) レコードに参照ロックをかける。
2) ルート索引ノードを参照ロックし下段の索引ノードをサーチする。
3) サーチされた索引ノードを参照ロックし上段の索引ノードをアンロックし下段の索引ノードをサーチする。
4) 3)を繰り返し最下段索引ノードまでたどりつくと、リーフノード内をサーチする。
5) チェック対象のレコードをポイントするレコード識別子(RID)の存在チェックを行う。
6) 最下段索引ノードとレコードをアンロックし、対象レコードをチェック完了とする。
7) 1)〜4)のロックが競合した場合は対象レコードのチェックを未処理（ロックは解除）とする。
8) 1)〜7)を全レコードのチェック処理が完了するまで繰り返す。

図-1に処理方式イメージ、図-2にロック期間を示す。

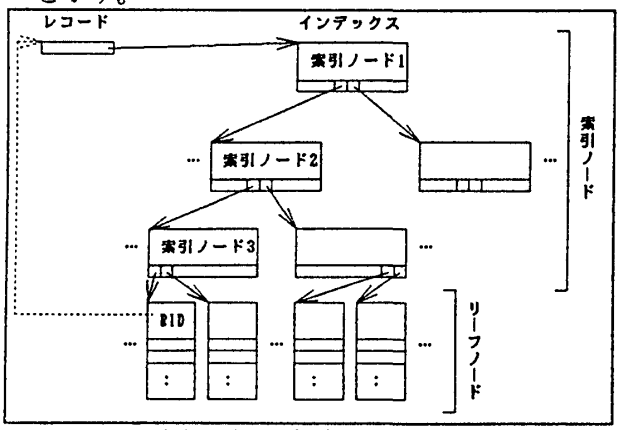

図-1 処理方式イメージ

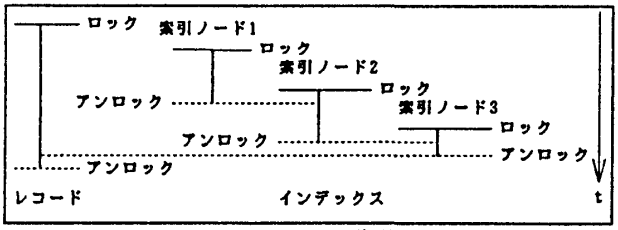

図-2 ロック期間

② インデックス → レコード相関チェック

1) ルート索引ノードを参照ロックし下段の索引ノードをサーチする。
2) サーチされた索引ノードを参照ロックし上段の索引ノードをアンロックし下段の索引ノードをサーチする。
3) 2)を繰り返し最下段索引ノードまでたどりつくと、リーフノード内RIDを取得する。
4) リーフノード内RIDのポイントするレコードを参照ロックする。
5) レコードの存在チェックを行う。
6) レコードと最下段索引ノードをアンロックし、対象レコードをチェック完了とする。
7) 1)〜4)のロックが競合した場合は対象レコードのチェックを未処理（ロックは解除）とする。
8) 1)〜7)を全リーフノード内RIDのチェック処理が完了するまで繰り返す。

図-3に処理方式イメージ、図-4にロック期間を示す。

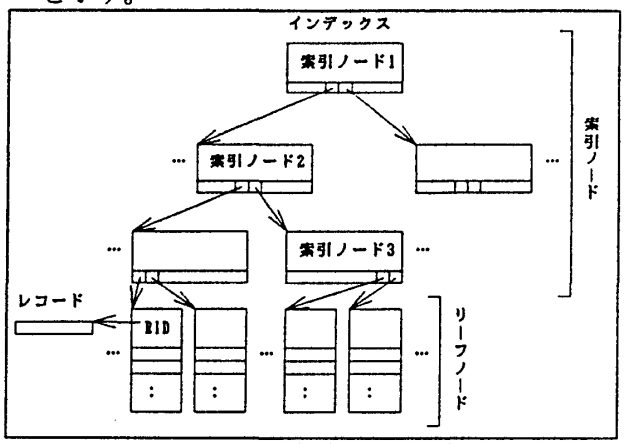

図-3 処理方式イメージ

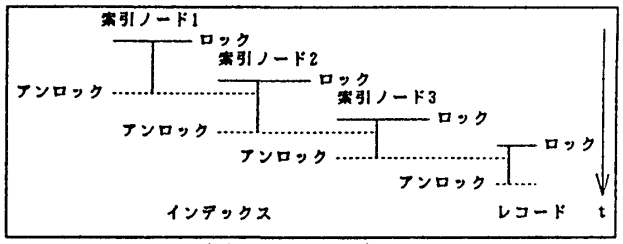

図-4 ロック期間

本方式により、ロック期間はインデックス更新を伴うトランザクションと同程度なため、DB整合性チェック処理とオンライントランザクションの競合時の影響（コンカレンシの低下）はオンライントランザクション同士が競合した場合と同じ程度で行うことが可能となりオンライントランザクションを止めることなく、動的DB状態でDB全体の整合性をチェックすることができる。

4．おわりに

本稿では、DBの局所的なブロックの整合性チェックを積み上げることにより、オンライントランザクションを止めることなく、動的DB状態でDB全体の整合性をチェックする方式を示した。