# 業務情報を用いたデータベース性能診断手法

6P−3

鬼塚 真,　大久保 成隆,　関根 純

NTT 情報通信網研究所

## 1 はじめに

DB設計では実世界を実体などの概念を用いてモデル化する手法がしばしば用いられるが、大規模なDBではその結果のモデルをそのまま用いてDB化することは性能に悪影響を及ぼすことが多い。そこで、性能を考慮してモデルを見直すことが重要となっている。RDBを例にとるとデータのアクセス単位であるテーブルを、性能を考慮して設計する手法（性能設計）は重要な課題である。

これまでの性能設計に関する研究[FKS]は、詳細情報の収集可能な詳細設計段階での定量的評価方法について主に行なわれてきた。しかし、この手法では評価結果によるテーブル再構成の手戻りが大きく、むしろ基本設計段階で支援する手法の方がDB設計者にとっての効果は大きい。

そこで筆者らは、DB設計の専門家が行なっている基本設計段階でのテーブル設計の手法を分析してノウハウ化し、それを用いてテーブルアクセス形態などに基づいた定性的評価による性能診断を行なうための手法を考案した。

2章でテーブル設計のノウハウとそのノウハウを計算機で処理する判定式について述べる。3章で、本手法の判定結果を実システムで専門家が行なった設計結果と比較し、その有効性を示す。

## 2 基本設計段階でのテーブル設計の性能診断手法

RDBにおけるテーブル設計のノウハウを社内の20システムについて分析した結果、以下の2つの観点に整理できる。

- テーブルの結合処理を減らす。
- 1テーブルへのアクセスの集中を減らす。

前者を実現するには、複数テーブルを1つのテーブルに統合するテーブル統合の手法が用いられ、後者を実現するには1つのテーブルを複数のテーブルに分解するテーブル分解の手法が用いられる。以下ではテーブル統合について、そのノウハウとそれを処理する判定式について述べる。

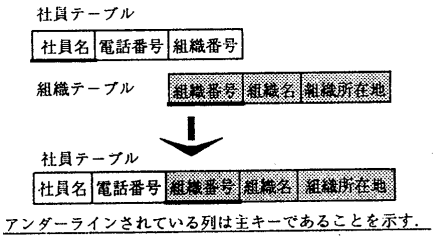
### 2.1 テーブル統合のノウハウ

基本設計段階におけるテーブル統合を更に調査して、次の3パターンを用いてテーブルを統合していることが分析できる。

**テーブルの完全統合** 多くの業務において、2テーブルが同時にアクセスされる率が高い場合、1つのテーブルに統合する(図1)。

**テーブルの部分コピー** 多くのオンライン処理業務において、テーブル1がアクセスされる場合、テーブル2の列も結合処理によりアクセスされる率が高い時、その列をテーブル1へコピーする。ただし、コピーするテーブル2への更新は少ない必要がある(図2: テーブル1は社員テーブル, テーブル2は組織テーブル)。

**繰り返しの復活** 多くの業務において、テーブル1をアクセスする時のみテーブル2をアクセスする場合、テーブル2の列をテーブル1へ展開する(図3: テーブル1は受注テーブル, テーブル2は商品テーブル)。




アンダーラインされている列は主キーであることを示す.

図 1: テーブルの完全統合の例

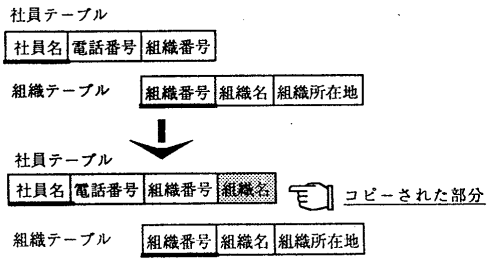
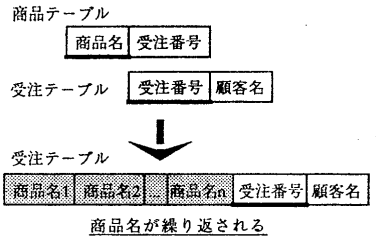



図 2: テーブルの部分コピーの例




商品名が繰り返される

図 3: 繰り返しの復活の例

### 2.2 テーブル統合の判定式

上記のノウハウは定性的な記述なので計算機処理に適していない。そこで基本設計段階でも収集可能なデータである「業務からテーブルへのアクセス形態」を用いて統合候補か否かを判定する尺度を表す式$S$（同時アクセス率）を考案した。

$$S(T_i,T_j) = \frac{G(T_i,T_j)}{H(T_j)}$$

$$\text{ただし、} H(T_i) = \sum_{k=0}^{n-1} ex(T_i,J_k),$$

$$G(T_i,T_j) = \sum_{k=0}^{n-1} ex(T_i,J_k) * ex(T_j,J_k),$$

---

An Evaluation Method of Database Performance using Table Access Knowledge

Makoto Onitsuka, Naritaka Ohkubo, Jun Sekine

NTT Network Information Systems Laboratories

$T_k$ : $k$ 番目のテーブル,
$J_k$ : $k$ 番目の業務,
$n$ : テーブルの個数,
$ex(T_k, J_l)$: テーブル $T_k$ と業務 $J_l$ の間のアクセスの有無を返す関数.

この同時アクセス率は、任意の業務に対してテーブル $T_i$ がアクセスされるとき同時にテーブル $T_j$ がどれほどアクセスされているかの度合を表す。よって同時アクセス率が高い程、統合する優先度が高い候補になる。

この同時アクセス率を用いてテーブルを統合するか否かの判定条件を与える。判定式は、統合のための必須条件である主判定式と、主判定式を補足するための補助判定式から構成される。以下に示す。ただし、$Ai, Bi, Ci$ は候補数を絞るための閾値である。

#### 2.2.1 主判定式

**テーブルの完全統合** $(S(T_i,T_j) > A1)$ and $(S(T_j,T_i) > A1)$
を満たすテーブル $T_i, T_j$ を統合する候補とする。

意味: テーブル $T_j$ がアクセスされる場合、テーブル $T_i$ の列がアクセスされる率が高く、かつその逆も成り立つ。

**テーブルの部分コピー** $S(T_i,T_j) > B1$
を満たす $T_i$,$T_j$ を、$T_i$ の列を $T_j$ へコピーする候補とする。

意味: テーブル $T_j$ がアクセスされる場合、テーブル $T_i$ の列がアクセスされる率が高い。

**繰り返しの復活** $S(T_i,T_j) > C1$
を満たす $T_i$,$T_j$ を、$T_i$ の列を $T_j$ へ展開する候補とする。

意味: テーブル $T_j$ がアクセスされる場合、テーブル $T_i$ の列がアクセスされる率が高い。

#### 2.2.2 補助判定式

補助判定式で用いられる関数は以下のように定義される。

$F_{T_j}(T_i)$: $0 \leq k \leq n$ なる $S(T_k,T_j)$ の中での $S(T_i,T_j)$ の偏差を返す関数,
$ac(T_i, J_j)$: テーブル $T_i$ と業務 $J_j$ の間のアクセス形態を返す関数 (アクセス形態とは { 参照, 更新, 削除 } である),
$gm(J_i)$: 業務 $J_i$ の業務形態を返す関数 (業務形態とは { オンライン, バッチ } である),
$A2, B2, B3, C2, C3$ は候補の足きりを行なうための値.

**テーブルの完全統合** $(F_{T_j}(T_i) > A2)$ and $(F_{T_i}(T_j) > A2)$

意味: テーブル $T_j$ がアクセスされる場合、テーブル $T_i$ がアクセスされる率が $T_i$ 以外のテーブルがアクセスされる率に比べ突出し、かつその逆も成り立つ。

**テーブルの部分コピー**

$$\sum_{k=0,ac(T_i,J_k)=\text{参照}}^{n-1} > (\sum_{k=0}^{n-1}(ex(T_i,J_k))) * B2 \text{ and}$$

$$\sum_{k=0,gm(J_k)=\text{オンライン}}^{n-1} (ex(T_i,J_k) * ex(T_j,J_k) > B3$$

意味: テーブル $T_i$ へのアクセスが参照である率が高く、かつテーブル $T_i, T_j$ へのアクセスを持つ業務はその形態がオンライン処理である率が高い。

**繰り返しの復活**

$F_{T_i}(T_j) > C3$ and
$\forall T_k.(k \neq j)$ and $(k \neq i)$ and$(S(T_i,T_k) < C4)$

意味: テーブル $T_j$ がアクセスされる条件で、テーブル $T_i$ がアクセスされる率が $T_i$ 以外のテーブルがアクセスされる率に比べ突出している。更にテーブル $T_j, T_i$ 以外の任意のテーブルがアクセスされる場合、同時に $T_i$ がアクセスされる率が小さい。

## 3 実システムにおける判定式の評価

本章では実用化されている 2 システムについて完全統合・繰り返しの復活それぞれについて、専門家の設計結果と 2 章の判定式による結果を比較した (図 4)。なお、閾値は固定的に使うのではなく、出力された候補の内容により変化させて使う。具体的には、候補数が一定以下となるように閾値を設定して足きりを行ない、評価値の大きい方から順に DB 設計者が統合の可否を判定する。評価値が低くなっても DB 設計者が統合と判断する事例がまだ現れる場合には閾値を下げる。意味ある統合の候補がなくなったら検討を打ち切る。

評価対象システムの規模

| | 業務数 | テーブル数 | 項目数 |
|---|---|---|---|
| システム 1 | 77 | 64 | 3573 |
| システム 2 | 146 | 75 | 822 |

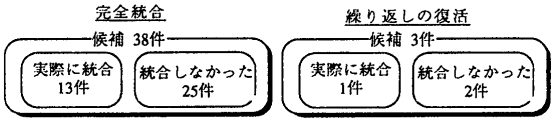



図 4: 評価結果

図 4 より、専門家が統合した実例が全て 2 章の判定式の候補に含まれることが分かる。

また、閾値による絞り込みを止めた場合、完全統合は 1096 件、繰り返しの復活は 379 件の候補が出る。この結果と図 4 を比較すると、本手法により充分に候補が絞り込まれていることが分かる。よって、本手法は DB 設計者が効率良く統合の候補を判断するのに有効であることが分かった。

また、実際に統合が行なわれなかった候補についての原因は、1. 統合により 1 テーブルへのアクセスが集中 (19 件)、2. 統合によりテーブル容量が増大 (6 件)、3. 別の統合パターンを選択 (1 件)、4.DB 以外の手法でテーブルを実現 (1 件) と分析できた。

1. については、テーブルの分解についてのノウハウに相当するため、これから検討を行なう必要がある。2. は容量の大きいテーブルを候補から外すノウハウを設定すれば十分であると考えられる。3. は複数の統合パターンが条件に合致した場合、どちらを選択すべきかの優先順位づけが今後の課題である。一方、4.5. についてはそれぞれの知識をノウハウ化する手法そのものが未解決なため、今後の課題である。

## 4 おわりに

本稿では、基本設計段階で適用可能な性能診断手法を考案した。また、本手法を実システムに適用することによりその有効性を確かめた。著者らは現在、本手法を DB 設計支援ツール [大久保] へ組み込み中である。今後は種々のシステムへ適用し閾値の決定を行ない、またテーブル分解についてのノウハウなど 3 章で未解決の問題を解決する考えである。

## 参考文献

[FKS] S. Finkelstein, M. Schkolnick, and P.Tiberio "Phiysical Database Design for Relaional Databases" *ACM Trans. Database Syst.*, Vol.13, No.1, pp.91-128(March 1988).

[大久保] 大久保, 町原, 関根, 中川 "DB 設計支援ツール DBprompt のアーキテクチャ" 情報処理学会第 44 回全国大会,1991.