リレーショナル・データベース上での統計データ管理の一方式†

中村 仁之輔†† 田中 豪†† 織田 敬三†††

本論文では，リレーショナル・データベース上で統計データの管理を効率的に実現するアレイ検索法について述べる．統計データベースの特徴に，データが調査・測定されたデータを分類するためのカテゴリ属性とカテゴリ属性の組合せに対応するサマリ属性に分類できる点がある．本特徴を利用した手法で，カテゴリ属性とサマリ属性を別々に管理し，計算により両者の対応付けを行うアレイ・リニアライゼーションが，既に提案されている．筆者らはアレイ・リニアライゼーションをリレーショナル・データベース管理システムに取り込み，統計データを管理する方法としてアレイ検索法を提案する．本論文では，アレイ検索法の実現方式について，アレイ・リニアライゼーションの取込み方法，リレーショナル・データベース言語の拡張法，および処理速度向上法を示し，最後に評価を行い，アレイ検索法の有効性を示す．

1. まえがき

リレーショナル・データベース (RDB) 管理システムは，データを表形式で扱う高度な問合せ言語を備えており，ユーザ・インタフェースが優れている，あるいは，業務処理プログラムの作成が簡易である，等の理由により，現在急速に普及しつつある．この RDB 上で統計データ管理を行う場合，(1)統計データベース特有の性質からくる効率上の問題，(2)統計処理に必要な機能上の問題が指摘されている[1]．

効率上の問題は，統計データが分類(カテゴリ)属性と各分類に対応する集約(サマリ)属性を持ち，特にカテゴリ属性に対する重複が大きいことが原因でデータベース容量の増加を招き，またこれによりアクセス効率の劣下を招くことである．機能上の問題は，標準的な RDB 問合せ機能には解析等の統計処理機能が存在しないため，業務プログラムの負担が大きいことである．ただし，後者の問題は既存の統計処理パッケージの利用等が考えられ，本稿では扱わないこととする．

効率上の問題を解決する手法として，データがカテゴリ属性とサマリ属性に分類できる特徴を利用し，格納効率，アクセス効率を向上させる方法であるアレイ・リニアライゼーション[2]が提案されている．アレイ・リニアライゼーションは，カテゴリ属性とサマリ属性を別々に管理しておき，その対応付けを計算のみで行う手法である．この方法をリレーショナル・データベース管理システム (RDBMS) に取り込むためには①柔軟性のあるテーブル構成法，②リレーショナル言語との融和性の良い言語インタフェース，③処理効率の良いアクセス・パスの実現法，が課題となる．筆者らは，これらを解決するための実現方式としてアレイ検索法を考案した．

本論文では，アレイ検索法における上記課題の解決方法およびその評価について述べる．2章でアレイ・リニアライゼーションの概要を示し，3章でアレイ検索法の実現方式を示す．具体的には，RDBMS におけるアレイ・リニアライゼーションの取込み方法として，テーブル構成法，言語インタフェースおよび検索処理時間を短縮するインデックス・サーチ法を提案する．最後の4章でアレイ検索法の評価を行い，本手法の有効性を明らかにする．

2. アレイ・リニアライゼーションの概要

2.1 統計データベースの特徴

統計データベースの特徴をデータの内容の観点および電子計算機上でのデータ利用特性の観点の二つの観点から整理すると以下のようになる[1),3)~7)]．

(1) データの内容に依存した統計データベースの特徴

① データは調査・測定されたデータを分類するためのカテゴリ属性（地域，性別等）とカテゴリ属性の組合せに対応するサマリ属性に分けられ，統計表中にカテゴリ属性値は重複して格納されている（図1参照）．

② カテゴリ属性の理論的な組合せに対応する実際のサマリ属性値がデータとして存在するとは限らない．

† A Method of Statistical Data Management on Relational Databases by JINNOSUKE NAKAMURA, TAKESHI TANAKA (NTT Communications and Information Processing Laboratories) and KEIZO ODA (NTT Data Communications Sector).

†† NTT 情報通信処理研究所
††† NTT データ通信事業本部

(カテゴリ属性)			(サマリ属性)
地　域	産業分類	年	就業者数
東　京	総　数	1970	5,670,685
東　京	総　数	1975	5,619,964
〃	〃	1980	5,650,100
〃	第一次産業	1970	59,306
〃	〃	1975	42,879
〃	〃	1980	34,400
〃	第二次産業	1970	2,202,464
〃	〃	1975	1,928,483
〃	〃	1980	1,795,300
〃	第三次産業	1970	3,396,436
〃	〃	1975	3,618,620
〃	〃	1980	3,795,900
神奈川	総　数	1970	2,643,063
〃	〃	1975	2,897,375
〃	〃	1980	3,151,800
〃	第一次産業	1970	105,254
〃	〃	1975	75,719
〃	〃	1980	70,100
・	・	・	・
・	・	・	・
・	・	・	・
・	・	・	・
・	・	・	・
・	・	・	・

図 1　カテゴリ属性とサマリ属性の例
Fig. 1　An example of category and summary attributes.

(2)　電子計算機上でのデータの利用特性からみた統計データベースの特徴

①　検索はカテゴリ属性に対する条件により，サマリ属性の集合を求めることが多い．また，検索条件はあるカテゴリ属性に対して値の範囲を指定（例えば，地域として関東地方に相当する東京，神奈川，…を指定）するものが多い．

②　データの変更は一般に少ない．ただし，データの追加は周期的に行われる．

2.2　アレイ・リニアライゼーション

カテゴリ属性の値は，同一カテゴリ内では重複して格納する必要はないため，それぞれのカテゴリ属性ごとに同一カテゴリ属性値はただ一つのみ格納しておく．この状態で，それぞれのカテゴリ属性についてカテゴリ属性値の種類数を要素数とする配列を考え，各カテゴリ属性の組合せを多次元配列と考える．各カテゴリ属性のカテゴリ属性値の配列内でのポジション（何番目の要素か）を組み合わせると，多次元配列内での任意の一つのポジションが確定できる．このポジションは多次元配列を一次元配列に変換した場合のポジションに変換することができ，この変換後のポジションに，各カテゴリ属性値の組合せに対応するサマリ属性値を配置しておくことにより，カテゴリ属性とサマリ属性の対応をとることができる．多次元配列の要素ポジションの組合せを一次元の要素ポジションに変換するアルゴリズムがアレイ・リニアライゼーションであり，以下のように表現できる．

$$\begin{aligned} &j_1 * N_2 * N_3 * \cdots * N_k + \\ &\quad j_2 * N_3 * \cdots * N_k + \\ &\qquad \vdots \\ &\qquad j_{k-1} * N_k + \\ &\qquad\quad j_k \end{aligned}$$

j_i：第 i 次元のカテゴリ属性の要素ポジション
N_i：それぞれのカテゴリ属性の配列要素数
k：カテゴリ属性数（次元数）

図 2 にアレイ・リニアライゼーションの例を示す．ここでは，$k=3$，N_1（地域）$=4$，N_2（産業分類）$=4$，N_3（年）$=3$ で，$j_1=0$（東京），$j_2=3$（第三次産業），$j_3=1$（1975）を計算すると，10 の値が結果として得られ，対応するサマリ属性値が 10 のポジションに配置されていることを示している．

3.　アレイ検索法

アレイ検索法を実現する上での課題は以下のとおりである．

(1)　アレイ・リニアライゼーションのためのテーブル構成法

(2)　リレーショナル言語としての言語インタフェース

(3)　(1)，(2)に対する効率的なアクセス法の実現

以下，それぞれの解決法を示す．

3.1　テーブル構成法

アレイ・リニアライゼーションを RDBMS に取り込む場合のテーブル構成法に関する課題に以下の二つがある．

(1)　アレイ・リニアライゼーションで求まった値（ポジション）の利用法

(2)　カテゴリ・データとサマリ・データの構成法

(1)については，物理的な表現，すなわちアレイ・リニアライゼーションで求まった値に対し一定の演算（例えば，タプル長を掛け合わせる）を行った結果がサマリ属性値の格納位置を示す形で実現する方法がある．しかし，RDBMS でアレイ・リニアライゼーショ

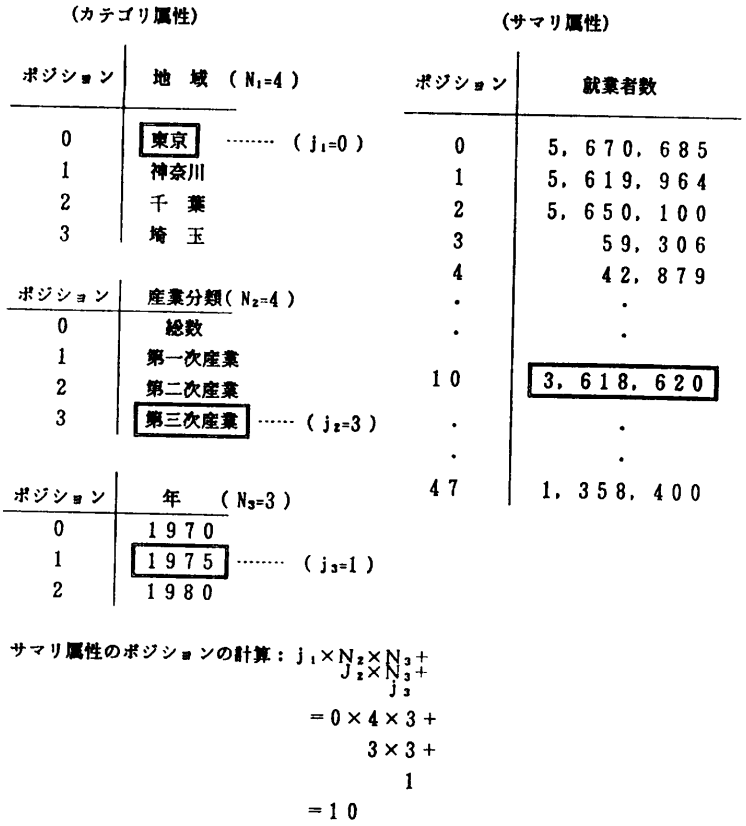


図 2 アレイ・リニアライゼーションの例
Fig. 2 An example of array linearization.

ン用に物理的格納位置を意識した格納手法を採った場合，通常のテーブルの格納空間とは別にアレイ・リニアライゼーション用に特別な格納空間が必要になり，複雑さが増す．そのため，論理的な表現，すなわちアレイ・リニアライゼーションにより求まった値をサマリ属性用のテーブル（サマリ・テーブル）中のタプルに保持しておき，その値に基づいたアクセスを行う方法を採る．ここで，高速アクセスのために，一般のRDBMS でサポートされているインデックスを利用し，サマリ・テーブル中のアレイ・リニアライゼーションで求まった値をキー値（サマリ・キー値と呼ぶ）とするインデックス経由のアクセスを実現する．また，サマリ・テーブルはインデックスの値に基づいたクラスタリング格納（タプルの近接格納）を実現する．インデックス・アクセスの詳細については，3.3 節で議論する．

(2)については，RDBMS 内部ではカテゴリ属性とサマリ属性を別々に管理するが，利用者プログラム（統計データ検索プログラム等）に対しては一連の統計データ・モデル（例えば一つのテーブル）として見せる方法も考えられる．しかし統計データの参照，解析では統計データ・モデルは一つとは限らず，種々の統計データ・モデル[7]を構築できる必要があるため，利用者プログラムの柔軟性の観点から，カテゴリ属性については統計データ検索プログラム側で独自に管理できることが重要である．そこで，アレイ・リニアライゼーションを実現するための固定的なデータ構造を設定し，統計データ検索プログラムとインタフェースをとる方法を採用する(図 3)．統計データ検索プログラムはこの基本構造をもとに，RDBMS への問合せを行い，エンド・ユーザに対しては必要に応じて一つのテーブルとして見せるインタフェースを設定する．

以下，図 3 のテーブル構成およびアクセス手順の概要を述べる．

[A] テーブル構成

以下に示す 3 種類のテーブルより構成される．

(a) カテゴリ辞書テーブル

カテゴリ属性の種別，各カテゴリ属性の要素数等より構成される．

(b) カテゴリ・テーブル

カテゴリ属性の種別ごとに，各カテゴリ属性の内容およびサマリ・キー値計算用の要素ポジションより構成される．

(c) サマリ・テーブル

カテゴリ属性の組合せに対応するサマリ属性とカテゴリ・テーブルの要素ポジションをもとに計算されるサマリ・キーより構成される．また，サマリ・キーに対してユニーク・インデックスが設定されている．

ここで，(a)，(b) のテーブルについては，この基本構造以外の任意の構成としてよく，最低限サマリ・テーブルにアクセス可能な情報（各カテゴリ属性の要素数，カテゴリ属性値の要素ポジション）が得られれば良い．

[B] アクセス手順

〈step 1〉 カテゴリ辞書テーブルよりカテゴリ属性を検索する．

〈step 2〉 カテゴリ属性に対応するカテゴリ・テーブルより指定されたカテゴリ属性値，要素数を検索す

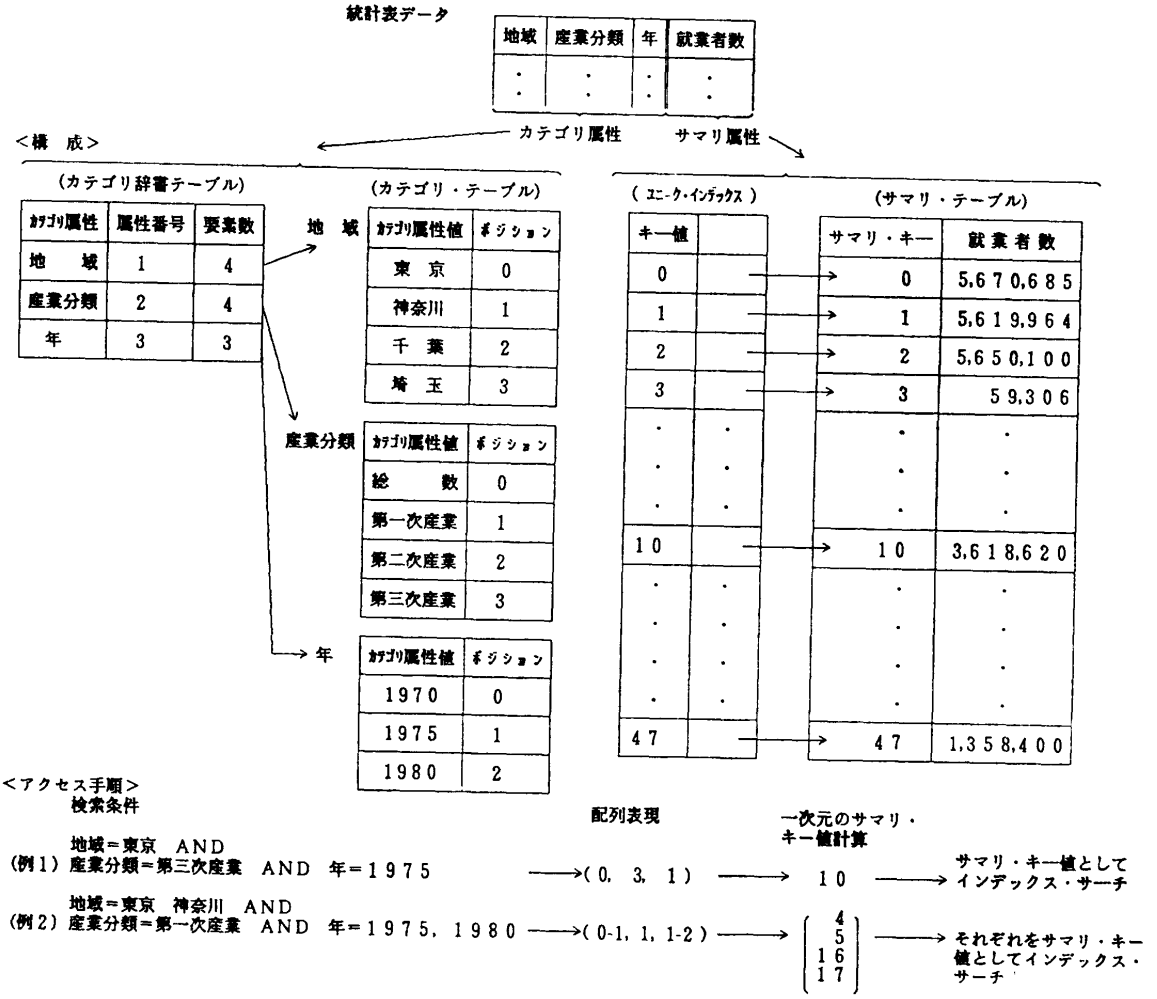


図 3 アレイ検索法の概要
Fig. 3 Array retrieval method.

る.

〈step 3〉 step 2 で求められたカテゴリ属性値，要素数よりアレイ・リニアライゼーションにより必要数分のサマリ・キー値を計算する.

〈step 4〉 step 3 で求められたすべてのサマリ・キー値によりサマリ・テーブルのインデックス・サーチを行い，該サマリ属性を検索する.

カテゴリ辞書テーブル，カテゴリ・テーブルは RDB のテーブルとして実現してもよいが，容量が少ないためメモリ上への展開が可能である.

本方法は，カテゴリ属性数の増減等カテゴリ属性とサマリ属性のマッピングに変更が生じるようなカテゴリ・テーブルの変更がある場合には，再編成（サマリ・キーの再割付け）が必要となるため，そのような変更が少ない応用分野に適している．サマリ・データの追加は，サマリ・キー値を付与したサマリ・データをサマリ・テーブルに追加するのみで良い.

3.2 言語インタフェース

アレイ検索用の言語は統計データ検索プログラムに対して提供することを想定し，SQL (Structured Query Language)[8] を対象に機能拡張を行う.

主たる考慮点は，既存言語との親和性を保ちつつ

(a) RDBMS内部の最適化機構によりサマリ・キーに対するアクセス機構（インデックス）が選ばれること，また，

(b) 統計データ検索プログラムに有用かつ効率的な問合せインタフェースであること，である.

拡張項目は次の四つである.

① 検索条件式で指定できる関数（アレイ・リニアライゼーション）の導入

② 出力項目として指定できる関数（逆アレイ・リニアライゼーション）の導入

③ ②に対するソート出力の拡張

④ ②に対するグループ化の拡張

SQL 仕様に追加した仕様を以下に示す．

（1） 関数1（アレイ・リニアライゼーション）

図3に示す基本構造を基に，カテゴリ・テーブルから得られる k 個の (N_i, j_i) 対 $(1 \leqq i \leqq k)$ を引数としてサマリ・キー値を導出する関数を導入する．これをアレイ関数と呼ぶ．本関数はサマリ・テーブルへのアクセス用条件式としてWHERE 文節で記述する．

［アレイ関数］

基本形：:=LIN $(N_1, j_1/N_2, j_2/\cdots/N_k, j_k)$

上記基本形より，2.2 節で示したアレイ・リニアライゼーションのアルゴリズムに従ってサマリ・キー値を求める．

WHERE 文節では，IN 述語の拡張形として指定する．

カラム名 IN アレイ関数

ここで，カラム名はサマリ・テーブルのサマリ・キー・カラム名を参照する．

また，2.1 節で述べたように，サマリ・キー値については，1回の指定でサマリ属性の集合を検索することが多いため，これを可能とするため，カテゴリ属性値の範囲指定ができるように拡張する．

N_i, j_i の組の表現として以下の表現を可能とする．

拡張形1：$N_i, j_{i1}, j_{i2}, \cdots, j_{il}$ $(l \leqq N_i)$ 離散値

拡張形2：$N_i, j_{i1}-j_{il}$ $(l \leqq N_i)$ 連続値

拡張形3：N_i (j_i の省略) すべての値

本関数が現れると RDBMS はアクセス経路として一意にサマリ・キー・カラムに対するインデックスを利用する．

（2） 関数2（逆アレイ・リニアライゼーション）

検索したサマリ属性値を最終的にエンド・ユーザに返すとき，統計データ検索プログラムではカテゴリ属性値の付加を必要とする．これを可能とするために

（例1）

［条件］地域＝東京，神奈川，産業分類＝第一次産業，年＝1975，1980の就業者数を年，地域の昇順で出力せよ．

［検索条件式］

```
SELECT  ILI（サマリ・キー・カラム／4，4，3），就業者数
FROM    サマリ・テーブル
WHERE   サマリ・キー・カラム IN
        LIN（4，1／4，1－3／3，1）
ORDER BY 3 ASC，1 ASC
        FOR ARRAY
```

［検索結果］

第一カテゴリ	第二カテゴリ	第三カテゴリ	就業者数
0	1	1	59，306
1	1	1	・
0	1	2	・
1	1	2	・

⇩

統計検索プログラムで，返却された各カテゴリ属性の要素ポジションを基に，カテゴリ・テーブルから，カテゴリ属性値を得て，統計表を出力する．

⇩

地域	産業分類	年	就業者数
東京	第一次産業	1975	59，306
神奈川	第一次産業	1975	・
東京	第一次産業	1975	・
神奈川	第一次産業	1975	・

（例2）

［条件］地域＝神奈川の産業分類別の就業者数の平均値を出力せよ．

［検索条件式］

```
SELECT  ILI（サマリ・キー・カラム／4，4，3），
        AVG（就業者数）
FROM    サマリ・テーブル
WHERE   サマリ・キー・カラム IN LIN（4，1／4／3）
GROUP BY 1，2 FOR ARRAY
```

［検索結果］

第一カテゴリ	第二カテゴリ	第三カテゴリ	平均就業者数
1	0	－	2，897，413
1	1	－	83，691
1	2	－	・
1	3	－	・

⇩

統計検索プログラムで，返却された各カテゴリ属性の要素ポジションを基に，カテゴリ・テーブルから，カテゴリ属性値を得て，統計表を出力する．

⇩

地域	産業分類	平均就業者数
神奈川	総数	2，897，413
神奈川	第一次産業	83，691
神奈川	第二次産業	・
神奈川	第三次産業	・

図4 検索条件式の例

Fig. 4 Examples of search condition.

は，現在処理しているサマリ属性値がどのカテゴリに属するかの情報を返却する機能が必要である．そこで，サマリ・キー・カラム名と k 個の N_i $(1 \leqq i \leqq k)$ を引数として，検索したサマリ属性から逆にその k 個（k 次元）のカテゴリ属性の各要素ポジションを返却する関数を導入する．これを逆アレイ関数と呼ぶ．本関数は SELECT 文節で指定する．

［逆アレイ関数］

基本形：:=ILI（カラム名/N_1, N_2, …, N_k）

上記基本形を基に 2.2 節のアルゴリズムから j_i $(1 \leqq i \leqq k)$ を求める．

（3） ソート出力

カテゴリ属性による出力順指定を可能とするため，ORDER BY 文節を拡張する．

ORDER BY 整数 [ASC/DESC]
[, 整数 [ASC/DESC]]
…
FOR ARRAY

FOR ARRAY を指定した ORDER BY 文節は，アレイ関数および逆アレイ関数を用いたサマリ属性の検索に対してのみ指定可能である．後者の制約は SQL との親和性を考慮し，整数を SELECT 文節中に指定される逆アレイ関数の引数に現れるカテゴリ属性の配列要素数の並びの出現順に対応させたことによる．

［例］ SELECT ILI（カラム名/N_1, N_2, …, N_i, …, N_k）
⋮
ORDER BY i ASC
FOR ARRAY

（4） グループ化

カテゴリ属性によるグループ化を可能とするため，GROUP BY 文節を拡張する．

GROUP BY 整数 [, 整数] …
FOR ARRAY

FOR ARRAY を指定した GROUP BY 文節は ORDER BY 文節と同様，アレイ関数および逆アレイ

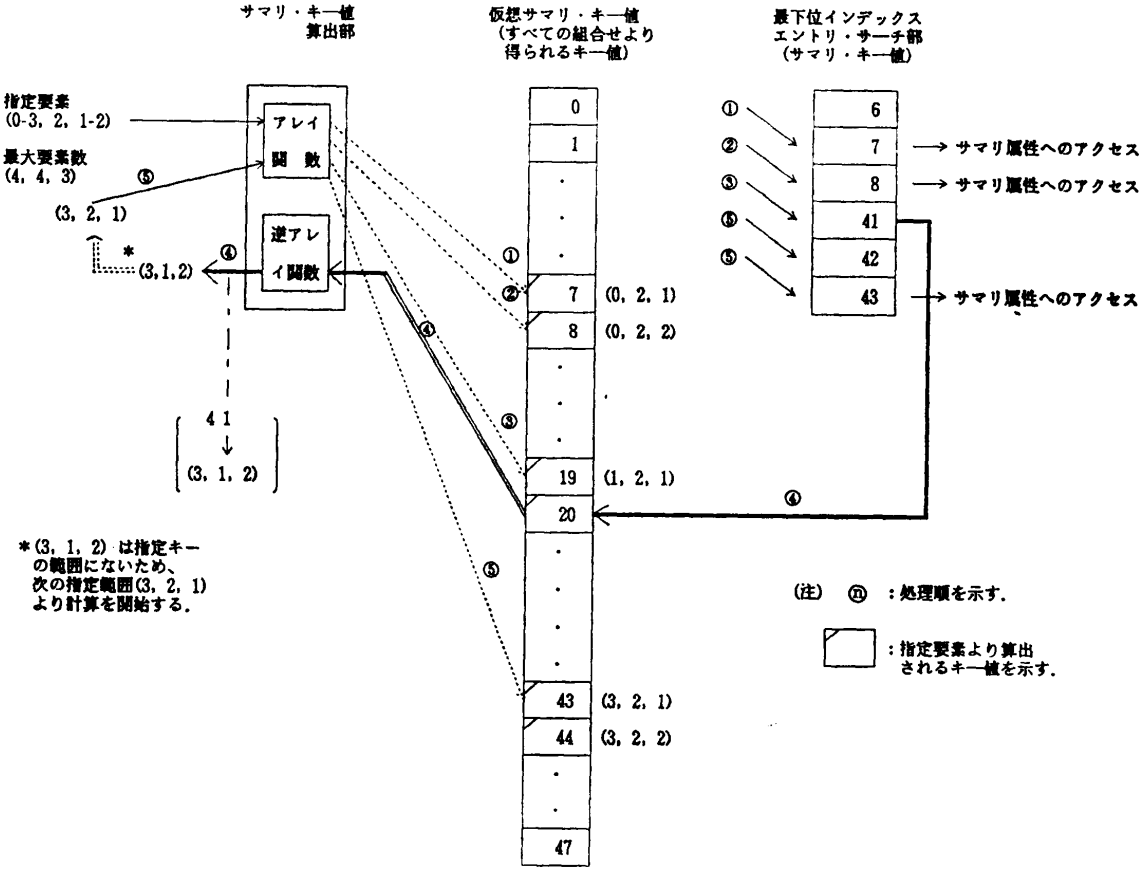


図 5 インデックス・サーチ方式の概要
Fig. 5 Index search method.

関数を用いたサマリ属性の検索に対してのみ指定可能であり，整数は逆アレイ関数の引数に現れるカテゴリ属性の配列要素の並びの出現順に対応する．

拡張した SQL 仕様による検索条件式の例を図 4 に示す．

3.3 インデックス・サーチの高速化手法

アレイ検索法では，サマリ・テーブルへのアクセスは B-tree 形式のインデックス経由とすることを想定している．2.1 節で述べたように，統計データベースの特徴は，カテゴリ属性の論理的な組合せに対する実際のサマリ属性値が存在するとは限らない点と，検索があるカテゴリ属性に対する範囲指定で行われることが多い点であり，このようなサマリ属性に対するアクセス効率の向上策が必要である．考慮点としては，① サマリ・キー値計算の結果を使ってアクセスする場合，無効なキー値計算を極力回避すること，また，② 範囲指定に対して，一つのキー値ごとにルート経由でアクセスするとオーバヘッドが大きいため，これを回避することである．そこで，実存するサマリ・キー値しかインデックス上には存在しない点に着目して，インデックス・アクセスとキー値計算を組み合わせ，かつ，開始キーのみルート経由で，その後は最下位インデックスのみをサーチするアルゴリズムを考えた（図 5）．

図 5 に示すようにサマリ・キー値算出部と最下位インデックス・エントリ・サーチ部を考える．

〈step 1〉 サマリ・キー値算出部では，アレイ関数により指定されたキー値（各カテゴリ属性ごとに要素ポジションが範囲で指定されたもの）に基づいてサマリ・キー値を昇順に一つずつ計算し，出力する．

〈step 2〉 インデックス・サーチ部では出力されたサマリ・キー値と最下位インデックス・エントリとの比較を行い，一致すればサマリ属性へのアクセスを行う（①，②）．算出されたサマリ・キー値がインデックス上にないとき（このとき，次のエントリのキー値より小さい），step 3 に行く（③）．

〈step 3〉 インデックス上の次のエントリのキー値より各次元の要素ごとの値を逆アレイ関数により求める（④）．次に，それぞれの要素について，指定キー値の範囲に含まれるかを判定し，含まれていればそのサマリ・キー値を開始キー値として step 1 の処理を続ける．含まれていなければ指定キー値より算出されるサマリ・キー値のうち該サマリ・キー値より大きくかつ最も近い値を開始キー値とし，step 1 の処理を続ける（⑤）．

4. 評　　価

本章ではアレイ検索法の定量評価として従来方式との比較評価を行う．従来方式としては，アレイ検索法を取り込まない RDB 上で統計データ管理を行う二つの方式，シングル・カラム・インデックス方式，マルチ・カラム・インデックス方式を取り上げる．

シングル・カラム・インデックス方式とは，統計表をそのまま一つのテーブルにし，カテゴリ属性に相当するカラムそれぞれにインデックスを設定する方式であり，従来の RDB 上で統計データ管理を行う場合に一般的に採用されている方法である．ただし，この場合も，DB 容量が大きくなる場合は，格納効率，アクセス効率を考慮し，カテゴリ属性はコード化されて格納される．マルチ・カラム・インデックス方式とは，全カテゴリ属性をマルチ・キーとするインデックスを一つ設定する点を除いてシングル・カラム・インデックス方式と同様である．

評価は，アレイ検索方式，シングル・カラム・インデックス方式，マルチ・カラム・インデックス方式それぞれについて，6 種類の検索パターンに対し，指定されるカテゴリ属性数をパラメータとして，検索時の I/O 回数を比較することによって行った．なお，ここでの評価は，RDBMS 内部の性能に関するものであり，統計データ検索プログラムにより行われる加工・編集に関する性能は対象外としている．また，I/O バッファの効果については，規則的なアクセスに対してのみ考慮し，ランダムなアクセスに対しては，考慮に入れていない．

(1) 評価データベース・モデル

表 1 に示すデータをもとにする[9)~11)]．表 1 は，カテゴリ属性数が i の場合，カテゴリ属性番号 1 から i 番

表 1 評価データベース・モデル
Table 1 Summary of database model.

カテゴリ属性番号	カテゴリ要素数	実データ存在率
1	3,300	1.0
2	48	0.5
3	3	0.5
4	100	0.4
5	18	0.3
6	4	0.075
7	10	0.075
8	12	0.075
9	30	0.06
10	20	0.03

目までのカテゴリ属性が存在することを示している．また，カテゴリ属性番号 i の実データ存在率 (E_i) は，統計データがカテゴリ属性番号1から i 番目までで構成されているとしたときの実存サマリ・キー数 (S_k) と論理サマリ・キー数との比率であり，以下の式で表現される．

$$E_i=\frac{S_k}{CV_1\times CV_2\times\cdots\times CV_i}$$

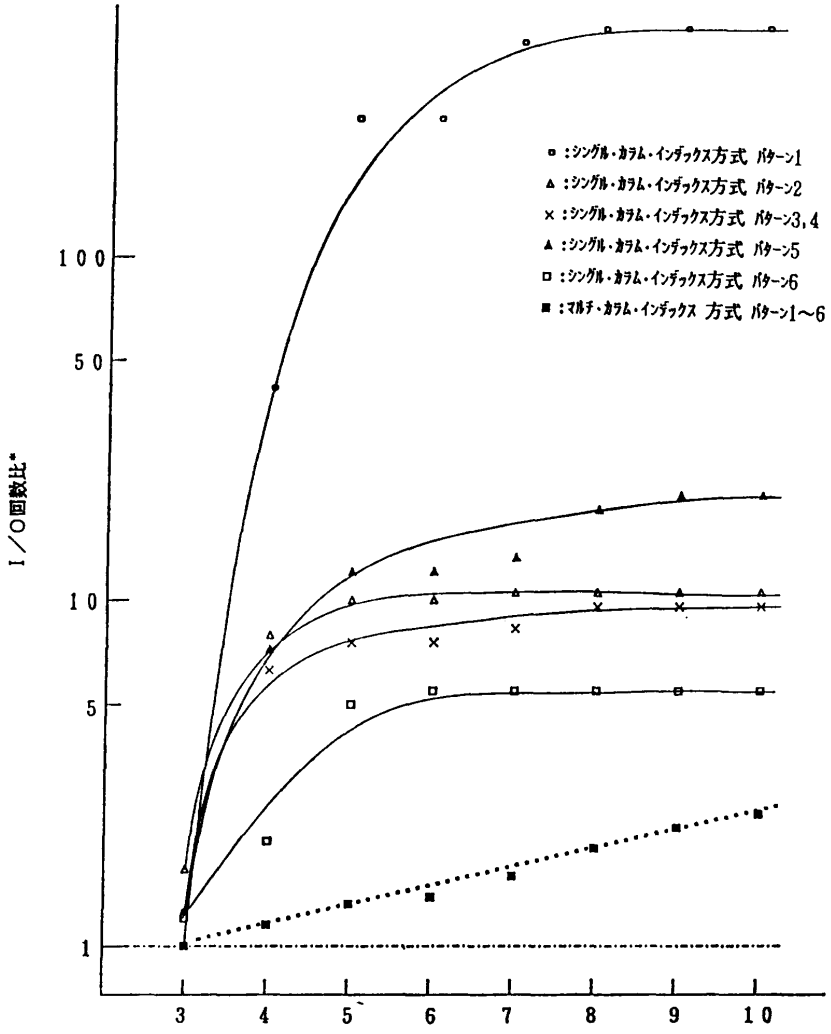
CV_i：カテゴリ属性番号 i のカテゴリ要素数

$1\leqq S_k\leqq CV_1\times CV_2\times\cdots\times CV_i$

ここで，アレイ検索方式ではサマリ・テーブルを構成する．シングル・カラム・インデックス方式，マルチ・カラム・インデックス方式では，カテゴリ属性を

表 2　問合せモデル
Table 2　Summary of query model.

カテゴリ属性番号	指定範囲					
	パターン1	パターン2	パターン3	パターン4	パターン5	パターン6
1	1	10	100	1,000	1	3,300
2	2	5	10	48	24	2
3	1	2	1	1	1	1
4	2	10	50	10	20	2
5	2	3	5	10	10	2
6	4	4	4	4	4	4
7	1	2	3	1	1	1
8	1	1	1	3	1	1
9	2	3	3	10	5	2
10	2	2	10	2	2	2



*：アレイ検索方式での I/O 回数を1としたときの相対値

図 6　I/O 回数比
Fig. 6　Ratio of I/O count.

コード化し，コード化されたカテゴリ属性とサマリ属性から構成されるテーブルとする．また，テーブル・タプルの配置は，アレイ検索方式ではサマリ・キーの値によって，シングル・カラム・インデックス方式では第一カテゴリ・カラム（一番目のカテゴリ属性に対応するカラム）の値によって，マルチ・カラム・インデックス方式では，マルチ・カラム・キー値によってクラスタリング格納が行われている．

（2） 問合せモデル

各カテゴリの指定範囲を変化させた6種類の問合せモデルを設定した（**表2**）．表2で，指定範囲とは，アレイ関数の拡張形2における，$j_{i1}-j_{i2}$ で指定した要素ポジション数を表す．また，各パターンにおいて，カテゴリ属性はカテゴリ属性番号の1から順に連続して指定される．カテゴリ属性番号1から i までを指定したとき，カテゴリ属性数 i の検索という．

シングル・カラム・インデックス方式への問合せは，任意の一つのインデックス中の対象キーを検索し，その後テーブル中のデータにアクセスし他のカラムに対する条件を判定するものとする．

（3） I/O 回数の評価

アレイ検索方式と単一テーブル方式のI/O回数比を**図6**に示す．

図6より，以下のことがわかる．

① 全体的に，アレイ検索方式が優れている．また，シングル・カラム・インデックス方式とマルチ・カラム・インデックス方式について見ると後者の方式が優れている．

② シングル・カラム・インデックス方式では，検索パターンによらず，少ないカテゴリ属性数では，アレイ検索方式とのI/O回数に大きな差はないが，カテゴリ属性数の増加に従い，アレイ検索方式とのI/O回数比が増加するとともに，その比が一定値に近づく．

③ シングル・カラム・インデックス方式では，第一カテゴリ範囲をすべて指定した場合は，アレイ検索方式とシングル・カラム・インデックス方式とのI/O回数比は小さくなっている．ただし，②の傾向は変わらない．

以上の現象は以下の理由による．

（a） アレイ検索方式では，アレイ関数の中で一意なサマリ・キー値を計算しており，サマリ・テーブルへの余分なアクセスがない．一方，シングル・カラム・インデックス方式では，インデックスによる絞り込みが一つのカテゴリ属性に限定されるため，アレイ検索方式に比べてサマリ・テーブルへのI/O回数が多くなる．また，マルチ・カラム・インデックス方式では，アレイ検索方式と同様，サマリ・テーブルへの余分なアクセスはないため，シングル・カラム・インデックス方式に比べて，I/O 回数は少ない．しかし，この方式はアレイ検索方式に比べて，キー長が大きくなるため，インデックスのサイズが大きくなる分インデックスへの I/O 回数が多くなる（以上①）．

（b） 第一カテゴリ・カラムでクラスタリングされているシングル・カラム・インデックス方式は，（a）で述べたサマリ・テーブルへの余分なアクセスによるI/O 回数がカテゴリ属性数の増加とともに，追加されたカテゴリの要素数にほぼ比例して増大する．一方，サマリ・キー値でクラスタリングされているアレイ検索方式では，カテゴリ属性数の増加とともに増加するI/O 回数は，対象となるサマリ・タプル数の増加の割合にほぼ比例する．このことから，両者のI/O回数比は，カテゴリ属性数の増加とともに，追加されたカテゴリ属性の指定範囲に対する該カテゴリ属性の要素数の比にほぼ比例して増加する．ただし，本評価で用いたデータベースモデルのように，カテゴリ属性番号の大きいカテゴリ属性の要素数が少ない場合は，そのカテゴリ属性の一つの値に対するタプルが同一ページに格納されるため，上記の比が1になり，カテゴリ属性数の増加とともに，両者の比が一定に近づく（以上②）．

（c） 条件に第一カテゴリが指定されず，第一カテゴリの全範囲が検索の対象となった場合，アレイ検索方式では最下位インデックスを全サーチし，インデックスへのI/O回数が増加するため，シングル・カラム・インデックス方式との性能差は小さくなる（以上③）．

表1で示した実データ存在率は，従来のインデックス・サーチ方式と新たに提案したインデックス・サーチ方式との比較を行う場合に関連するパラメータであり，両方式の差は，I/O バッファの効果がある場合は，キー値計算による CPU での処理時間のみであり，I/O 回数への評価に影響を与えない．

以上より，本評価データベース・モデルでは，アレイ検索方式が優れていることがわかる．ここで，データベース・モデルのカテゴリ属性数，カテゴリ要素数を変化させた場合を考える．この場合も，アレイ検索方式と性能の良いマルチ・カラム・インデックス方式とを比較すると，（a）で述べたように，マルチ・カラム・インデックス方式のキー長が長くなるため，インデックスのサイズが大きくなる分インデックスへの

I/O 回数が増えるため，アレイ検索方式が優れている．

5. むすび

本論文では，カテゴリ属性の内容変更が少ない統計データをリレーショナル・データベース上で効率的に管理できるアレイ検索法の実現方式を示し，その有効性を明らかにした．

要約すると，

① アレイ・リニアライゼーションを RDB 上で実現する際のテーブル構成法と SQL 拡張法の提案，

② 上記に対する効率的アクセスを実現するインデックス・サーチ法の提案，

③ I/O 回数比較による本方式の有効性の提示，

を行った．

今後の課題としては，インデックス・サーチにおいて，カテゴリ属性の内容変更は少ない点に着目し，事前にインデックス・キー（サマリ・キー）の分布特性を把握し，その特性を反映したサーチ方法の確立がある．

謝辞　本研究の機会を与えていただいた知能処理研究部橋本昭洋部長，日頃ご指導いただいている情報処理研究部森道直主幹研究員（前データベース研究室室長），情報蓄積方式研究室吉田清室長（前データベース研究室主幹研究員）に深謝します．

参考文献

1) Shoshani, A.: Statistical Databases: Characteristics, Problems, and Some Solutions, *Proc. 8th VLDB*, pp. 208-222 (1982).
2) Eggers, S. J. and Shoshani, A.: Efficient Access of Compressed Data, *Proc. 6th VLDB*, pp. 205-211 (1980).
3) Turner, M. J. et al.: A DBMS for Large Statistical Databases, *Proc. 5th VLDB*, pp. 319-327 (1979).
4) 中村仁之輔，田中　豪，織田敬三：RDB による統計データベース処理効率化のための一方式，第 28 回情報処理学会全国大会論文集，pp. 679-680 (1984).
5) 中村仁之輔，田中　豪，織田敬三：アレイ検索法におけるインデックス・サーチの一方式，第29回情報処理学会全国大会論文集，pp. 887-888 (1984).
6) 中村仁之輔，田中　豪，織田敬三：RDB 上でのアレイ検索法の評価，第 30 回情報処理学会全国大会論文集，pp. 969-970 (1985).
7) 佐藤英人：リレーショナルデータベースと統計データベース，*Computer Today*, No. 7, pp. 18-21 (1985).
8) ISO/TC97/SC21/WG3 N117 and ANSI X3H2-86-26, "Database Language SQL", March (1986).
9) 総理府統計局：国勢調査報告 (1981).
10) 総理府統計局：家計調査年報 (1982).
11) 総理府統計局：社会生活統計指標 (1982).

(昭和 61 年 4 月 18 日受付)
(昭和 62 年 6 月 11 日採録)

中村仁之輔（正会員）

昭和 30 年生．昭和 52 年愛媛大学工学部電子工学科卒業．同年日本電信電話公社（現，NTT）横須賀電気通信研究所入所．以来，オンライン情報検索システム，データベース管理システムの研究実用化に従事．現在，NTT 情報通信処理研究所主任研究員．昭和 60 年学術奨励賞受賞．電子情報通信学会会員．

田中　豪（正会員）

昭和 25 年生．昭和 48 年九州大学工学部電気工学科卒業．同年日本電信電話公社（現，NTT）横須賀電気通信研究所入所．以来，データベース管理システムの研究実用化に従事．現在，NTT 情報通信処理研究所主幹研究員．電子情報通信学会会員．

織田　敬三（正会員）

昭和 21 年生．昭和 44 年慶応義塾大学工学部電気工学科卒業．同年日本電信電話公社入社．以来，電気通信研究所にて情報検索システム，データベース管理システムの研究実用化に従事．現在，NTT データ通信事業本部企画部主任技師．電子情報通信学会会員．