(12) 特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2009 年8 月6 日 (06.08.2009)

PCT

(10) 国際公開番号
WO 2009/095981 A1

(51) 国際特許分類:
G06F 17/30 (2006.01) *G06F 17/21* (2006.01)
G06F 12/00 (2006.01)

(21) 国際出願番号: PCT/JP2008/051185

(22) 国際出願日: 2008 年1 月28 日 (28.01.2008)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(71) 出願人 (米国を除く全ての指定国について): 株式会社 ターボデータラボラトリー (TURBO DATA LABORATORIES INC.) [JP/JP]; 〒2310004 神奈川県横浜市中区元浜町三丁目２１番地２ Kanagawa (JP).

(72) 発明者; および

(75) 発明者/出願人 (米国についてのみ): 古庄 晋二 (FURUSHO, Shinji) [JP/JP]; 〒2310004 神奈川県横浜市中区元浜町三丁目２１番地２ Kanagawa (JP).

(74) 代理人: 吉田 聡 (YOSHIDA, Satoshi); 〒2330001 神奈川県横浜市港南区上大岡東２－２４－４ Kanagawa (JP).

(81) 指定国 (表示のない限り、全ての種類の国内保護が可能): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) 指定国 (表示のない限り、全ての種類の広域保護が可能): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MT, NL, NO, PL, PT, RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

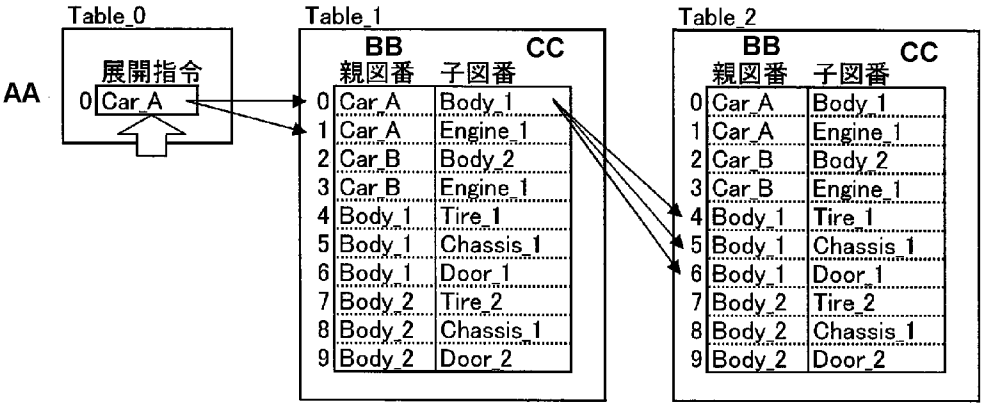
[続葉有]

(54) **Title:** METHOD AND DEVICE FOR BUILDING TREE-STRUCTURED DATA FROM TABLE

(54) 発明の名称: 表からツリー構造データを構築する方法及び装置

[図9]

図9



AA SPREAD INSTRUCTION
BB PARENT DRAWING NUMBER
CC CHILD DRAWING NUMBER

(57) **Abstract:** Tree-structured data formed by a plurality of nodes each expressed by an attribute name and an attribute value is expressed by a virtual record arrangement in which one record corresponds to one node and each record is identified by a record order number expressing the record arrangement order. Each record includes a parent relationship field for storing a record order number of a parent node of the node corresponding to the record, an attribute name field for storing an attribute name number which specifies an attribute name belonging to the node, and an attribute value field for storing an attribute value number which specifies an attribute value belonging to the node.

[続葉有]

WO 2009/095981 A1

添付公開書類:
— 国際調査報告書

(57) 要約: 各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データは、１個のレコードが１個のノードに対応し、各レコードがレコードの並び順を表すレコード順序番号によって識別される、仮想的なレコードの配列によって表現される。レコードは、当該レコードに対応するノードの親ノードのレコード順序番号を格納する親子関係フィールドと、当該ノードに属する属性名を指定する属性名番号を格納する属性名フィールドと、当該ノードに属する属性値を指定する属性値番号を格納する属性値フィールドとを含む。

明 細 書

表からツリー構造データを構築する方法及び装置

技術分野

[0001] 本発明は、表形式データからツリー構造データを構築する方法及び装置に係わり、特に、各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを、各レコードがレコードの並び順を表すレコード順序番号によって識別される仮想的なレコードの配列として、記憶装置上に構築する方法及び装置に関係する。

[0002] 本発明は、表構造の原データを記憶するメモリを備え、上記原データを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し上記メモリに格納する装置にも関係する。

[0003] さらに、本発明は、コンピュータに上記方法を実行させるためのプログラム、プログラムプロダクト、及び、プログラムを記録した記録媒体に関係する。

[0004] さらに、本発明は、各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データが、各レコードがレコードの並び順を表す上記レコード順序番号によって識別される、仮想的なレコードの配列として記録された記録媒体に関係する。

背景技術

[0005] 現在最も普及しているデータベースマネージメントシステム(DBMS)はリレーショナルデータベース(RDB)である。RDBは表形式データベースであるため、ツリー構造データの取り扱いを苦手としていることがよく知られている。

[0006] また、一般に、表形式データからツリー構造データを抽出(又は解釈)する作業は非常に厄介である。たとえば、対象とする表のレコードが、最上位項目から最下位項目までの複数個の項目により定義されている場合を考える。このとき、表形式データからツリー構造データを解釈するためには、

(1)最上位項目に同じ項目値を有するレコードをグループ分けし、

(2)上位項目に関して同じグループに分類されたレコードを、現在項目に関して同じ

項目値を有するレコード毎にグループ分けし、

(3)現在項目が最下位項目に達するまで(2)を繰り返す、

という操作が必要である。この操作は、表形式データがn行m列(レコード数＝n、項目数＝m)であるならば、O(n＊m)回の比較演算を必要とする。また、同じ項目名が多数回出現し、かつ、同じ項目値が多数回出現するので、これらの情報を記憶するために多量のメモリが必要である。さらに、同じ項目値が多数回出現するので、データ更新時に多数個の値を書き換えなければならない。

[0007] そこで、ツリー構造データを階層化された複数個の表形式データで記述する種々の手法が提案されている。

[0008] 第一に、一般的に木構造を持つ拡張マークアップ言語(XML)データをRDBに関連付けることは困難であるが、XMLデータの持つ自由度や柔軟性を大きく阻害することなく、XMLデータを表形式のデータ構造で表現する技術が提案されている(特許文献1)。特許文献1に記載されている技術では、XMLデータは、ルートエレメントから順にたどって各テーブルに格納され、格納されたデータには出現順序情報を付与される。各テーブルには、出現順序情報を含むダウングリップカラムと、格納されたデータと親子関係にある親データのダウングリップカラムの出現順序情報が割り当てられたアップグリップカラムとが作成されている。さらに、各テーブルには、ルートエレメントからの階層の深さを表すレベル値を格納するレベルカラムが作成されている。しかし、この技術では、XMLデータのスキーマ情報に基づいてデータの単位となるエレメント毎に1つのテーブルが作成され、テーブルにスキーマ情報に従ったデータが格納される。

[0009] 第二に、項目に属する項目値を含むレコードの配列として表現される複数の表形式データを結合し、深さの浅い表形式データのレコードから深さの深い表形式データのレコードまでを順次特定し、これにより、深さと、レコードを示す値とからなるツリー記述テーブルを作成することにより、ツリー構造データを複数の表形式データへ変換する技術も提案されている(特許文献2)。この例でも、深さ毎に1つの表形式データが存在している。

[0010] 特許文献1又は特許文献2において取り扱われている表形式データは、ツリー構造

データに変換することが意図されたデータである。たとえば、特許文献1では、ダウングリップカラム及びアップグリップカラムがテーブルの中に存在している。これに対して、近年、生産管理システムやトレーサビリティシステムで使用されている部品表と呼ばれるBOM(Bill of Materials)は、本来、ツリー構造データに変換されることが予定されていないが、データを分析したり、データを理解したりするとき、部品表をツリーデータ構造に展開すること(BOM展開)が必要とされる。たとえば、ストラクチャー型部品表では親部品図番と子部品図番の対が1つのデータとして格納されている。たとえば、親部品Aに子部品B1、B2及びB3が存在する場合、(A, B1)、(A, B2)及び(A, B3)の3組の対がストラクチャー型部品表に格納されている。さらに、子部品B1がC1及びC2という子部品を含むならば、(B1, C1)及び(B1, C2)という2組の対もまたストラクチャー型部品表に格納されている。

特許文献1:特開2004－178084号公報

特許文献2:国際公開第2004／038612号公報

特許文献3:国際公開第2005／088479号公報

特許文献4:国際公開第00／10103号公報

特許文献5:国際公開第03／040960号公報

発明の開示

発明が解決しようとする課題

[0011] 特許文献1に記載されている従来技術では、予めどのようなデータ構造が存在するかをスキャンによって、又は、文書型定義(DTD)のような定義によって知った上で、データを格納する格納先テーブルを作成する必要がある。よって、様々なデータ構造が現場で生成されるエディタなどの用途には、性能の面で不適当である。

[0012] また、特許文献1又は特許文献2に記載されているような、ツリー構造データを複数個の階層化された表形式データで記述する従来のデータ表現形式は、1個の表形式データで記述する手法より、比較演算回数が削減され、メモリ使用量が削減されているが、任意のトポロジーをもつツリーを追加することが困難であり、データ編集の自由度が低い。

[0013] よって、ツリー構造データを記述するデータ表現形式は、データ構造の動的な変化

に対応可能であり、高い編集の自由度を有することが好ましい。

[0014] ツリー構造データは、それ自体を解釈するための比較操作が削減され、それ自体を格納するためのメモリ使用量が削減され、ツリー構造の更新時に書き換えるべきデータ量が削減され、事前のデータ構造の定義を不要とすることが好ましい。

課題を解決するための手段

[0015] 本発明の一実施例によれば、各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを各レコードがレコードの並び順を表すレコード順序番号によって識別される仮想的なレコードの配列として記憶装置上に構築する装置であって、

階層構造を有する複数のテーブルを階層毎にマッチングキーによってマッチングさせ、親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成する手段と、

上記行番号対応表から親子関係がある上記親側のテーブル中の行番号及び上記子側のテーブル中の行番号を探索する手段と、

上記探索する手段によって見つけられた順に、上記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、上記仮想的なレコードの配列中の上記レコード順序番号によって識別される位置に、親子関係フィールド値として上記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、上記ノードの属性名フィールド値として上記子側のテーブル中の行に属する項目名を格納し、上記ノードの属性値フィールド値として上記子側のテーブル中の行に属する項目値を格納する手段と、

を備える装置が提供される。

[0016] 一実施例では、1個のレコードは1個のノードに対応していてもよい。階層構造を有する複数のテーブルは、ツリー構造データに展開されるべき表構造の原データである。また、マッチングキーはジョインキーでよい。ジョインキーを用いるマッチング、すなわち、ジョインは、原データからジョインテーブル、すなわち、中間データを生成する。ジョインテーブルは、親子関係にある原データ自体と、親子関係にある原データ間の結合関係を表現するようなテーブルである。

[0017] 本発明の一実施例によれば、上記装置は、上記ノードの属性名フィールド値が上記レコード順序番号の順番に格納されている配列から、上記レコード順序番号の順に上記属性名を一意に特定する属性名番号が格納されている属性名番号配列、上記属性名番号配列の要素の順番に上記属性名番号によって一意に特定される上記属性名が格納されている属性名配列、上記レコード順序番号の順に上記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、上記属性値番号配列の要素の順番に上記属性値番号によって一意に特定される上記属性値が格納されている属性値配列を作成する手段をさらに備える。

[0018] 本発明の一実施例によれば、上記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、上記行番号対応表を作成する手段が、最上位階層のテーブルとして、上記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のデータとして同一の上記ストラクチャー型部品表を利用する。

[0019] 本発明の一実施例によれば、上記行番号対応表を作成する手段が、上記マッチングキーとして、上記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、上記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する。

[0020] 本発明の一実施例によれば、階層構造を有する複数のテーブルを記憶するメモリを備え、上記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し上記メモリに格納する装置であって、

上記複数のテーブルを階層毎にマッチングキーによってマッチングさせるマッチング部と、

上記複数のテーブルの階層毎にマッチングした親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成する対応表作成部と、

上記行番号対応表から親子関係がある上記親側のテーブル中の行番号及び上記子側のテーブル中の行番号を探索する探索部と、

上記探索部によって見つけられた順に、上記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、上記仮想的なレコードの配列中の上記レコード順序番号によって識別される位置に、親子関係フィールド値として上記子側のテーブル

中の行に対応するノードの親ノードのレコード順序番号を格納し、上記ノードの属性名フィールド値として上記子側のテーブル中の行に属する項目名を格納し、上記ノードの属性値フィールド値として上記子側のテーブル中の行に属する上記原データに属する項目値を格納するレコード生成部と、

を備え、

上記ツリー構造データが、各レコードがレコードの並び順を表す上記レコード順序番号によって識別される、仮想的なレコードの配列として上記メモリに構築される装置が提供される。

[0021] 本発明の一実施例によれば、上記装置は、上記ノードの属性名フィールド値が上記レコード順序番号の順番に格納されている配列から、上記レコード順序番号の順に上記属性名を一意に特定する属性名番号が格納されている属性名番号配列、上記属性名番号配列の要素の順番に上記属性名番号によって一意に特定される上記属性名が格納されている属性名配列、上記レコード順序番号の順に上記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、上記属性値番号配列の要素の順番に上記属性値番号によって一意に特定される上記属性値が格納されている属性値配列を作成する値分離部をさらに備える。

[0022] 本発明の一実施例によれば、上記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、上記対応表作成部が、最上位階層の表構造の中間データとして、上記ツリー構造データのルートノードに対応する部品を選択し、その他の表構造の中間データとして同一の上記ストラクチャー型部品表を利用する。

[0023] 本発明の一実施例によれば、上記対応表作成部が、上記マッチングキーとして、上記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、上記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する。

[0024] 本発明の一実施例によれば、各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを各レコードがレコードの並び順を表すレコード順序番号によって識別される仮想的なレコードの配列として記憶装置上に構築する方法であって、

階層構造を有する複数のテーブルを階層毎にマッチングキーによってマッチングさせ、親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成するステップと、

上記行番号対応表から親子関係がある上記親側のテーブル中の行番号及び上記子側のテーブル中の行番号を探索し、探索によって見つけられた順に、上記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、上記仮想的なレコードの配列中の上記レコード順序番号によって識別される位置に、親子関係フィールド値として上記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、上記ノードの属性名フィールド値として上記子側のテーブル中の行に属する項目名を格納し、上記ノードの属性値フィールド値として上記子側のテーブル中の行に属する項目値を格納するステップと、

を備える方法が提供される。

[0025] 本発明の一実施例によれば、上記方法は、上記ノードの属性名フィールド値が上記レコード順序番号の順番に格納されている配列から、上記レコード順序番号の順に上記属性名を一意に特定する属性名番号が格納されている属性名番号配列、上記属性名番号配列の要素の順番に上記属性名番号によって一意に特定される上記属性名が格納されている属性名配列、上記レコード順序番号の順に上記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、上記属性値番号配列の要素の順番に上記属性値番号によって一意に特定される上記属性値が格納されている属性値配列を作成するステップをさらに備える。

[0026] 本発明の一実施例によれば、上記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、上記行番号対応表を作成するステップにおいて、最上位階層のテーブルとして、上記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のテーブルとして同一の上記ストラクチャー型部品表を利用する。

[0027] 本発明の一実施例によれば、上記行番号対応表を作成するステップにおいて、上記マッチングキーとして、上記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、上記子側のテーブルにおけるデータに対応する親側のデータに割

り当てられた順序番号との対を利用する。

[0028] 本発明の一実施例によれば、階層構造を有する複数のテーブルを記憶するメモリを備え、上記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し上記メモリに格納するコンピュータに、

上記複数のテーブルを階層毎にマッチングキーによってマッチングさせる機能と、

上記複数のテーブルの階層毎に、マッチングした親側のテーブルと子側のテーブルとの間で、親子関係がある行番号同士の行番号対応表を作成する機能と、

上記行番号対応表から親子関係がある上記親側のテーブル中の行番号及び上記子側のテーブル中の行番号を探索する機能と、

探索によって見つけられた順に、上記子側のテーブル中の行に割り当てられる上記レコード順序番号を決定する機能と、

上記ツリー構造データが、各レコードがレコードの並び順を表す上記レコード順序番号によって識別される、仮想的なレコードの配列として上記メモリに構築されるように、上記仮想的なレコードの配列中の上記レコード順序番号によって識別される位置に、親子関係フィールド値として上記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、上記ノードの属性名フィールド値として上記子側のテーブル中の行に属する項目名を格納し、上記ノードの属性値フィールド値として上記子側のテーブル中の行に属する項目値を格納する機能と、

を実現させるためのプログラムが提供される。

[0029] 本発明の一実施例によれば、上記プログラムは、上記ノードの属性名フィールド値が上記レコード順序番号の順番に格納されている配列から、上記レコード順序番号の順に上記属性名を一意に特定する属性名番号が格納されている属性名番号配列、上記属性名番号配列の要素の順番に上記属性名番号によって一意に特定される上記属性名が格納されている属性名配列、上記レコード順序番号の順に上記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、上記属性値番号配列の要素の順番に上記属性値番号によって一意に特定される上記属性値が格納されている属性値配列を作成する機能を上記コンピュータさらに実現させる。

[0030] 本発明の一実施例によれば、上記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、上記プログラムは、最上位階層のテーブルとして、上記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のテーブルとして同一の上記ストラクチャー型部品表を利用する機能を上記コンピュータにさらに実現させる。

[0031] 本発明の一実施例によれば、上記プログラムは、上記マッチングキーとして、上記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、上記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する機能を上記コンピュータにさらに実現させる。

[0032] 本発明の一実施例によれば、上記プログラムを記録したコンピュータ読み取り可能な記録媒体が提供される。

[0033] 本発明の一実施例によれば、階層構造を有する複数のテーブルを記憶するメモリを備え、上記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し上記メモリに格納するコンピュータに、

上記複数のテーブルを階層毎にマッチングキーによってマッチングさせる機能と、

上記複数のテーブルの階層毎に、マッチングした親側のテーブルと子側のテーブルとの間で、親子関係がある行番号同士の行番号対応表を作成する機能と、

上記行番号対応表から親子関係がある上記親側のテーブル中の行番号及び上記子側のテーブル中の行番号を探索する機能と、

探索によって見つけられた順に、上記子側のテーブル中の行に割り当てられるレコード順序番号を決定する機能と、

上記ツリー構造データが、各レコードがレコードの並び順を表す上記レコード順序番号によって識別される、仮想的なレコードの配列として上記メモリに構築されるように、上記仮想的なレコードの配列中の上記レコード順序番号によって識別される位置に、親子関係フィールド値として上記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、上記ノードの属性名フィールド値として上記子側のテーブル中の行に属する項目名を格納し、上記ノードの属性値フィールド値として上記子側のテーブル中の行に属する項目値を格納する機能と、

を実現させるためのプログラムプロダクトが提供される。

[0034] 本発明の一実施例によれば、各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを記録したコンピュータ読み取り可能な記録媒体であって、

上記ツリー構造データは、各レコードがレコードの並び順を表すレコード順序番号によって識別される、仮想的なレコードの配列によって表現され、

レコードは、上記レコードに対応するノードの親ノードのレコード順序番号を格納する親子関係フィールドと、上記ノードに属する属性名を指定する属性名番号を格納する属性名フィールドと、上記ノードに属する属性値を指定する属性値番号を格納する属性値フィールドとを含み、

上記ノードの親ノードが上記ノードに対応する上記レコードに含まれる上記親子関係フィールドに格納されている上記レコード順序番号によって特定され、

上記ノードに属する属性名が上記ノードに対応する上記レコードに含まれる上記属性名番号と上記属性名が上記属性名番号の順番に格納されている属性名配列とによって特定され、

上記ノードに属する属性値が上記ノードに対応する上記レコードに含まれる上記属性値番号と上記属性値が上記属性名番号の順番に格納されている属性値配列とによって特定される、

コンピュータ読み取り可能な記録媒体が提供される。

[0035] 本発明の一実施例によれば、上記レコード順序番号は、深さ優先探索又は幅優先探索の順で上記複数のノードのうちの各ノードに対応する上記レコードに割り当てられている。

発明の効果

[0036] 本発明の少なくとも1つの実施例によれば、ツリー構造データを解釈するための比較操作が削減され、メモリ使用量が削減され、ツリー構造の更新時に書き換えるべきデータ量が削減され、事前のデータ構造の定義を不要とするようなツリー構造データの構築が可能になる。

図面の簡単な説明

[0037] [図1]図1は、本発明の一実施形態によるツリー構造データを処理するコンピュータシステムのハードウェア構成を示すブロック図である。

[図2]図2は、ツリー構造データの例におけるノードの階層関係を示す図である。

[図3]図3Aは、本発明の一実施形態によるツリー構造データを記述する深さ優先によるデータ表現形式の説明図である。図3Bは、本発明の一実施形態によるツリー構造データを記述する幅優先によるデータ表現形式の説明図である。

[図4]図4Aは、本発明の一実施形態によるツリー構造データを記述する値が分離されたデータ表現形式の説明図である。図4Bは、本発明の一実施形態によるツリー構造データを記述する情報ブロック型のデータ表現形式の説明図である。

[図5]図5は、情報ブロックに基づくデータ管理機構を説明するための表形式データの一例を表す図である。

[図6]図6は、情報ブロック型のデータ表現形式の説明図である。

[図7]図7は、ストラクチャー型部品表の一例の説明図である。

[図8]図8は、ストラクチャー型部品表に埋め込まれている階層関係を示す図である。

[図9]図9は、ストラクチャー型部品表から作成された中間データと中間データ間のレコードの対応関係を示す図である。

[図10]図10Aは、ストラクチャー型部品表から生成された深さ優先形式によるツリー構造データの一例の説明図である。図10Bは、ストラクチャー型部品表から生成された幅優先形式によるツリー構造データの一例の説明図である。

[図11]図11Aは、ストラクチャー型部品表から生成された深さ優先形式によるツリー構造データ(情報ブロック型)の一例の説明図である。図11Bは、ストラクチャー型部品表から生成された幅優先形式によるツリー構造データ(情報ブロック型)の一例の説明図である。

[図12]図12Aは、ツリー構造データを記述する複数のテーブルの一例の説明図である。図12Bは、複数のテーブルで記述されたツリー構造データの階層関係を示す図である。

[図13]図13Aは、本発明の一実施形態によるレコード番号対応表の作成処理の説明図である。図13Bは、本発明の一実施形態によるレコード番号対応表の作成処理

の説明図である。

[図14]図14Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図14Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図14Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図15]図15Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図15Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図15Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図16]図16Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図16Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図16Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図17]図17Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図17Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図17Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図18]図18Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図18Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図18Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図19]図19Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図19Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図19Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図20]図20Aは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図20Bは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図である。図20Cは、本発明の一実施形態

による深さ優先方式に基づくツリー構造データ生成処理の説明図である。

[図21]図21Aは、統合されたツリー構造データを示す図である。図21Bは、値リストが分離されたツリー構造データを示す図である。図21Cは、情報ブロック型のツリー構造データを示す図である。

[図22]図22は、本発明の一実施形態による情報ブロックに基づくデータ形式への変換処理の説明図である。

[図23]図23は、本発明の一実施形態による値リストの共通化処理の説明図である。

[図24]図24は、本発明の一実施形態による項目値のリストを値番号と値リストに分離する処理の説明図である。

[図25]図25Aは、本発明の一実施形態によるジョインを利用するレコード番号対応表の作成処理の説明図である。図25Bは、本発明の一実施形態によるジョインを利用するレコード番号対応表の作成処理の説明図である。

[図26]図26Aは、本発明の一実施例によるジョインを利用するレコード番号対応表の作成処理の説明図である。図26Bは、本発明の一実施例によるジョインを利用するレコード番号対応表の作成処理の説明図である。

[図27]図27は、本発明の一実施形態によるツリー構造データの生成処理を適用した結果の説明図である。

[図28]図28は、本発明の他の実施形態によるツリー構造データの生成処理を適用した結果の説明図である。

[図29]図29は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[図30]図30は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[図31]図31は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[図32]図32は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[図33]図33は、本発明の一実施形態による幅優先方式のツリー構造データ生成処

理の説明図である。

[図34]図34は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[図35]図35は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理を用いて生成された情報ブロック型ツリー構造データの説明図である。

[図36]図36は、本発明の一実施形態によるツリー構造データの生成処理を適用した結果の説明図である。

[図37]図37は、本発明の他の実施形態によるツリー構造データの生成処理を適用した結果の説明図である。

符号の説明

[0038]

10	コンピュータシステム
12	CPU
14	RAM
16	ROM
18	ハードディスク
19	CD－ROM
20	CD－ROMドライバ
22	インタフェース
24	入力装置
26	表示装置
28	バス

発明を実施するための最良の形態

[0039] 以下、本発明を実施するための種々の形態を図面と共に詳細に説明する。

[0040] [コンピュータシステム構成]

図1は、本発明の一実施形態によるツリー構造データを処理するコンピュータシステムのハードウェア構成を示すブロック図である。図1に示すように、このコンピュータシステム10は、通常のコンピュータシステムと同様の構成であり、プログラムを実行することによりシステム全体および個々の構成部分を制御するCPU12と、ワークデータ

などを記憶するRAM(Random Access Memory)14と、プログラム等を記憶するROM(Read Only Memory)16と、ハードディスク等の固定記憶装置18と、CD－ROM19を駆動するためのCD－ROMドライバ20と、CD－ROMドライバ20や外部ネットワーク(図示せず)と接続された外部端子との間に設けられたインタフェース(I／F)22と、キーボードやマウスからなる入力装置24と、表示装置26とを備えている。CPU12、RAM14、ROM16、外部記憶装置18、I／F22、入力装置24および表示装置26は、バス28を介して相互に接続されている。

[0041] 本発明の一実施形態によるツリー構造データを処理するプログラムは、CD－ROM19に収容され、CD－ROMドライバ20によって読取られても良く、或いは、ROM16に予め記憶されていても良い。また、いったんCD－ROM19から読み出したものを、外部記憶装置18の所定の領域に記憶しておいても良い。或いは、上記プログラムは、ネットワーク(図示せず)、外部端子およびI／F22を経て外部から供給されるものであっても良い。

[0042] また、本発明の一実施形態による情報処理装置は、コンピュータシステム10にツリー構造データを処理するプログラムを実行させることにより実現される。

[0043] なお、以下の説明では、特に断らない限り、本発明の種々の実施形態は、コンピュータシステムの環境で実施され、動作の主体は、プログラム命令等のソフトウェアを実行するCPU等のプロセッサ、専用ハードウェア、又は、ファームウェア、或いは、それらの組み合わせである。

[0044] [ツリー構造データのデータ表現形式]

最初に、本発明の一実施形態によるツリー構造データのデータ表現形式を説明する。以下の説明では、一例として、XML形式では、

```
＜Story＞Harry Potter
  ＜Family＞Potter
    ＜Name＞James＜／Name＞
    ＜Name＞Lily＜／Name＞
    ＜Name＞Harry＜／Name＞
  ＜／Family＞
```

＜Family＞Black

＜Name＞Sirius＜／Name＞

＜／Family＞

＜／Story＞

と記述されるツリー構造データの例を考える。このXMLテキストの1行目は、属性名＝Storyと属性値＝Harry Potterとからなる属性を定義している。2行目は、属性名＝Familyと属性値＝Potterとからなる属性を定義している。

[0045] 図2はこのツリー構造データの例におけるノードの階層関係を示す図である。同図によれば、ルートノードは、属性名：属性値＝Story：Harry Potterという属性を保有するノードであり、ルートノードには、Family：Potterという属性を保有する子ノードと、Family：Blackという属性を保有する子ノードとが存在することがわかる。さらに、ノードFamily：Potterには、子ノードName：Jamesと、子ノードName：Lilyと、子ノードName：Harryとが存在する。一方、ノードFamily：Blackには、子ノードName：Siriusが存在する。ここで、ノードは、ノード属性名：属性値という表記によって指定されている。

[0046] 図3は、本発明の一実施形態によるツリー構造データを記述するデータ表現形式の説明図である。特に、図3Aは、図2に示されたツリー構造データの例における深さ優先でのデータ表現形式を示し、図3Bは幅優先でのデータ表現形式を示している。

[0047] ここで、深さ優先方式と幅優先方式について簡単に説明する。たとえば、図2のツリー構造データのノードをルートノードから順番に深さ優先方式で探索し、見つけられた順番にノードにノード識別子を付与する場合を考える。このとき、各ノードに付与されるノード識別子は、ルートノードのノード識別子を0とすると、

ノードStory：Harry Potterのノード識別子＝0

ノードFamily：Potterのノード識別子＝1

ノードName：Jamesのノード識別子＝2

ノードName：Lilyのノード識別子＝3

ノードName：Harryのノード識別子＝4

ノードFamily：Blackのノード識別子＝5

ノードName:Siriusのノード識別子＝6

となる。一方、図2のツリー構造データのノードを幅優先方式で探索し、同様に、ノードにノード識別子を付与すると、

ノードStory:Harry Potterのノード識別子＝0

ノードFamily:Potterのノード識別子＝1

ノードName:Jamesのノード識別子＝3

ノードName:Lilyのノード識別子＝4

ノードName:Harryのノード識別子＝5

ノードFamily:Blackのノード識別子＝2

ノードName:Siriusのノード識別子＝6

となる。すなわち、深さ優先探索とは、同じ世代のノードより子ノードを優先するノード探索を表し、幅優先探索とは、子ノードより同じ世代のノードを優先するノード検索を表している。このようなツリー構造データの記述法は特許文献3に記載されている。

[0048] ここで、本明細書中での配列の記法について説明する。一般に、配列Aは、添字をiとすると、配列の要素がA[i]のように表記できるが、図面中では、配列は、配列の要素A[i]は、実線で囲まれた領域内に示され、要素A[i]と要素A[i＋1]の境界は点線で示されている。また、要素A[i]の添字iが要素A[i]の左側に示されている。また、配列の添字iは0から始まる整数で表されている。

[0049] 図3A、Bを再び参照すると、ツリー構造データは、1個のレコードが1個のノードに対応し、各レコードがレコードの並び順を表すレコード順序番号によって識別される、レコードの配列によって表現されている。なお、ノードとレコードが1対1で対応しているので、レコード順序番号は上述のノード識別子と一致している。

[0050] レコードは、レコードに対応するノードの親ノードのレコード順序番号を格納する親子関係フィールドTopologyと、当該ノードに属する属性名を指定する属性名フィールドTitleと、当該ノードに属する属性値を指定する属性値フィールドValueとを含む。たとえば、レコード順序番号＝iであるレコードRecord[i]の各フィールドの要素は、Topology[i]、Title[i]及びValue[i]によって表される。

[0051] 深さ優先による図3Aの例では、レコード順序番号＝0のレコードの各フィールドの

要素は、

Topology[0]＝－1

Title[0]＝Story

Value[0]＝Harry Potter

となる。親子関係フィールドTopologyに、レコード順序番号として存在しない値、たとえば、－1が格納されているレコードは、このレコードが親ノードの存在しないノードに対応していること示している。よって、レコード順序番号＝0のレコードはルートノードに対応し、ルートノードの属性名がStoryであり、属性値がHarry Potterであることもわかる。

[0052] さらに、図3Aの例において、レコード順序番号＝3のレコードの各フィールドの要素は、

Topology[3]＝1

Title[3]＝Name

Value[3]＝Lily

である。よって、このレコードは、親ノードのレコード順序番号が1であるノードに対応し、かつ、ノードの属性名＝Name、ノードの属性値＝Lilyであることを示している。当該ノードの親ノードは、レコード順序番号＝1のレコードに対応するノード、すなわち、親ノードのレコード順序番号＝0、ノードの属性名＝Family、かつ、ノードの属性値＝Potterを保有するノードであることがわかる。

[0053] 一方、幅優先による図3Bの例においても、たとえば、レコード順序番号＝6のレコードの各フィールドの要素は、

Topology[6]＝2

Title[6]＝Name

Value[6]＝Sirius

である。よって、このレコードは、親ノードのレコード順序番号が2であるノードに対応し、かつ、ノードの属性名＝Name、ノードの属性値＝Siriusであることを示している。当該ノードの親ノードは、レコード順序番号＝2のレコードに対応するノード、すなわち、親ノードのレコード順序番号＝0、ノードの属性名＝Family、かつ、ノードの属性

値＝Blackを保有するノードであることがわかる。

[0054] このように、本発明の一実施形態によるツリー構造データを記述するデータ表現形式は、1つの表に統合されているので、非常に操作性に優れ、新しいツリーの追加又はツリーの削除に柔軟に対応し、データ編集の自由度が高い。

[0055] ここで、図3A及び3Bに示されているツリー構造データを記述するレコードの配列は、メモリ等の記憶装置上にこのままの形式で、すなわち、配列Topology、配列Title及び配列Valueとして記憶されている必要はなく、このレコードの配列を実質的に表現可能であればどのような表現でも構わない。このようにレコードの配列を等価的に表現できるデータは、本書中では、「仮想的な」レコードの配列と呼ばれることがある。

[0056] たとえば、レコードは、当該レコードに対応するノードの親ノードのレコード順序番号を格納する親子関係フィールドと、当該ノードに属する属性名を指定する属性名番号を格納する属性名フィールドと、当該ノードに属する属性値を指定する属性値番号を格納する属性値フィールドとによって定義することができる。そして、当該ノードの親ノードが当該ノードに対応するレコードに含まれる親子関係フィールドに格納されているレコード順序番号によって特定され、当該ノードに属する属性名が当該ノードに対応するレコードに含まれる属性名番号と属性名が属性名番号の順番に格納されている属性名配列とによって特定され、当該ノードに属する属性値が当該ノードに対応するレコードに含まれる属性値番号と属性値が属性名番号の順番に格納されている属性値配列とによって特定されるように構成することが可能である。

[0057] これは、属性名や属性値のような値を、値リストVLと値リストへのポインタ配列VNoとの組み合わせによって表現する考え方に基づいている。値リストVLと値リストへのポインタ配列との組み合わせは「情報ブロック」とも呼ばれ、特許文献4において提案されている概念である。

[0058] 図4A、Bは、この情報ブロックの概念を導入することにより記憶装置上に構築された、本発明の一実施形態によるツリー構造データのデータ表現形式の説明図である。同図の例は、深さ優先方式に従っている。図4Aは、属性名Titleが属性名番号配列Title__VNoと属性名配列Title__VLとに分離され、属性値Valueが属性値番号

配列Value__VNoと属性値配列Value__VLとに分離されたデータ表現形式の説明図であり、図4Bは図4Aの表現形式に基づいて導出された情報ブロック型のデータ表現形式の説明図である。たとえば、レコード順序番号iを指定すると、親子関係フィールドの値は、Topology[i]によって特定され、属性名は、Title__VL[Title__VNo[i]]によって特定され、属性値は、Value__VL[Value__VNo[i]]によって特定されることがわかるであろう。

[0059] [情報ブロックに基づくデータ管理機構]

図4A、Bのデータ表現形式がより良く理解されるように、情報ブロックに基づくデータ管理機構について簡単に説明する。

[0060] 図5は情報ブロックに基づくデータ管理機構を説明するための表形式データの一例を表す図である。この表形式データは、上述の特許文献4に提案されたデータ管理機構を用いることにより、コンピュータ内では図6に示されるようなデータ構造として、メモリ(たとえばRAM14)に記憶される。このデータ構造は、市販されているコンピュータ、たとえば、パーソナルコンピュータのハードウェア資源、特に、プロセッサ及びメモリを使用して大規模な表形式データの検索、ソート、集計等を実現するために提案された、コンピュータのメモリ上に置かれる表形式データのデータ構造であることに注意すべきである。

[0061] なお、「元の表形式データ中でレコードが収容されている位置を表す情報レコード番号(すなわち、原始レコード位置番号)」と「レコードの並び順を表す情報(すなわち、レコード順序番号)」とは明確に区別されるべきである。すべてのレコードには原始レコード位置番号が関連付けられている。この原始レコード位置番号は、データ項目に対応した項目値を含む個々のレコードを特定するために利用される仮想的な情報である。一般に、表形式データは、レコードが常に原始レコード位置番号の順番に配列されているとは限らない。たとえば、元の表形式データをある項目の項目値に関して昇順にソートすると、得られる表形式データのレコードの並び順は元の表形式データのレコードの並び順とは異なる。但し、元々の表形式データ中のレコードは、レコードが原始レコード位置番号の順番に並べられていることがあり、この場合には、原始レコード位置番号とレコード順序番号とが初期的に一致している。

[0062] 図6に示すように、表形式データの各レコードの並び順の番号(レコード順序番号)と、原始レコード位置番号は、レコード順序指定配列601(以下、この配列を「OrdSet」のように略記する。)によって対応付けられる。レコード順序指定配列601は、レコード順序番号の順に原始レコード位置番号を格納している。図6の例では、レコードは原始レコード位置番号の順番に並べられている。

[0063] 図6を参照すると、性別に関しては、表形式データのレコード順序番号＝0に対応する原始レコード位置番号は、配列OrdSet[0]から「0」であることがわかる。原始レコード位置番号が「0」であるレコードに関する実際の性別の値、即ち、「男」又は「女」は、実際の値が所定の順序(たとえば、昇順又は降順)に従ってソートされた値リストである項目値配列603(以下、項目値配列、すなわち、値リストを「VL」のように略記する。)へのポインタ配列である項目値番号配列602(以下、項目値番号配列、すなわち、ポインタ配列を「VNo」のように略記する。)を参照することによって取得できる。ポインタ配列602は、配列OrdSet601に格納されている原始レコード位置番号の順番に従って、実際の値リスト603中の要素を指し示すポインタを格納している。これにより、表形式データのレコード「0」に対応する性別の項目値は、(1)配列OrdSet601からレコード順序番号＝0に対応する原始レコード位置番号＝0を取り出し、(2)値リストへのポインタ配列602から原始レコード位置番号＝0に対応する要素「1」を取り出し、(3)値リスト603から、値リストへのポインタ配列602から取り出された要素「1」によって指し示される要素「女」を取り出すことにより取得できる。

[0064] 他のレコードに対しても、また、年齢及び身長に関しても同様に項目値を取得することができる。

[0065] このように表形式データは、値リストVLと、値リストへのポインタ配列VNoの組合せにより表現され、この組合せを、特に、「情報ブロック」と称する。図6には、性別、年齢及び身長に関する情報ブロックがそれぞれ情報ブロック608、609及び610として示されている。

[0066] 単一のコンピュータが単一のメモリ(物理的には複数であっても良いが、単一のアドレス空間に配置されアクセスされるという意味で単一のメモリ)を有するならば、単一のコンピュータは、当該メモリに、順序集合の配列OrdSet、各情報ブロックを構成す

る値リストVLおよびポインタ配列VNoとを記憶しておけばよい。しかしながら、本発明の種々の実施形態では、表形式データの操作は、複数台の演算ユニットにより並列的に実行することも可能である。

[0067] ［部品表（BOM）展開］

次に、生産管理システムにおける部品表（BOM）展開に関して本発明の一実施形態を説明する。以下の説明では、Car__Aという車とCar__Bという車の構成（又は、構成に対応した部品図番）が部品表の形で与えられている場合を考える。図7は、車の構成に関するストラクチャー型部品表の一例の説明図である。同図に示された部品表には、10組の親部品の図番（すなわち、親図番）と子部品の図番（すなわち、子図番）の対が格納されている。左端の0から9までの番号は、部品表のレコード番号を表している。部品表の親図番と子図番の対、すなわち、レコードは、このレコード番号によって識別される。たとえば、部品表の先頭行（すなわち、レコード番号＝0のレコード）には、親図番＝Car__Aと子図番＝Body__1の対が格納されている。また、レコード番号＝1のレコードには、親図番＝Car__Aと子図番＝Engine__1の対が格納されている。さらに、レコード番号＝4、レコード番号＝5及びレコード番号＝6のレコードには、親図番＝Body__1が格納されているので、それぞれの子図番は、Tire__1、Chassis__1及びDoor__1であることがわかる。このように、図7の部品表には、部品の階層関係が埋め込まれているので、特に、ストラクチャー型部品表とも呼ばれる。

[0068] 図8は図7の部品表に記述されている車の構成の階層関係を示す図である。本発明の一実施形態は、図7に示されている部品表（原データ）を、図8に示されているツリー構造データに変換（展開）する方法を提供する。この方法は、本発明の一実施形態では、図1に示されたコンピュータシステム10によって実行される。つまり、以下に述べる処理ステップは、コンピュータシステム10、特に、コンピュータシステム10中のCPU12により実行される。また、処理ステップにおいて生成されるデータ、表、配列は、それぞれ、メモリ（たとえば、RAM14）中に格納される。変換後のツリー構造データは、図3A、B又は図4A、Bを参照して説明したツリー構造データのデータ表現形式によって記述される。

[0069] ステップ1:展開したいデータの指定

部品表自体は、親子関係を記述しているが、たとえば、深さ優先順又は幅優先順に整列されているとは限らない。したがって、部品表をツリー構造データに変換するために、最初に、展開したいデータ(ツリー構造データのルートノード)が指定される。展開したいデータは、たとえば、図1に示されたコンピュータシステム10のハードディスク18、入力装置24に予め記憶され、又は、CD－ROMドライバ20を介してCD－ROM19から供給され、又は、I／F22を介して外部端子から供給されるなどの様々な形で与えられる。コンピュータシステム10のCPU12は、何れかの形で与えられたデータを展開したいデータとして指定する。本例では、Car__Aをツリー構造データに変換する場合を考える。

[0070] ステップ2:展開したいデータの表の作成

次に、第1階層の中間データとして、展開したいデータの表が作成される。具体的には、項目名＝展開指令、項目値＝Car__Aという1個のレコード(レコード番号＝0)からなる表が作成される。なお、中間データとして作成されるレコードは、レコード順序の入れ替えを想定する必要がないので、簡単のため、レコード順序番号がレコード番号と呼ばれることもある。また、項目名と項目値を併せて項目と呼ぶことがあるが、このとき、項目は項目名で区別される。よって、項目＝展開指令という表記は、項目名＝展開指令という項目を指している。

[0071] ステップ3:レコード番号対応表の作成(1回目)

第2階層の中間データとして、原データである部品表が準備される。そして、第1階層の中間データ(すなわち、親側のテーブル)と第2階層の中間データ(すなわち、子側のテーブル)との間で、共通化のためのキー項目(すなわち、マッチングキー)に関してマッチングが実行され、2つの階層間でレコード番号対応表が作成される。たとえば、第1階層の中間データ中のあるレコード(たとえば、レコードa)と、第2階層の中間データ中のあるレコード(たとえば、レコードb)が、当該キー項目に関して同一の項目値を有するならば、第1階層の中間データ中のレコードaと第2階層の中間データ中のレコードbの組み合わせがレコード番号対応表に登録される。なお、本書において、原データ又は中間データに含まれているレコードは、仮想的なレコード配列中のレ

コードと区別するために、「行」と呼ばれることがある。同様に、レコード番号が「行番号」と呼ばれ、レコード番号対応表が「行番号対応表」と呼ばれることがある。

[0072] このようなマッチングの一例はジョインによって実現することが可能である。ジョインとは、たとえば、特許文献5に記載されているように、情報ブロックを用いて記述されている2個の表形式データの間で、等価な項目（すなわち、ジョインキー）を見出し、等価な項目に関する情報ブロックに含まれる値リストを比較し、両方の表形式データで参照される値リストを共通化する技術である。ジョインを使ってレコード番号対応表を作成する方法は後述されている。

[0073] 本例では、第1階層の中間データの項目＝展開指令と、第2階層の中間データの項目＝親図番が共通化のためのキー項目として選択される。第1階層の中間データのレコード番号＝0のレコード（以下では、簡単のため、レコード番号＝NのレコードをレコードNと表記することがある）には、項目値＝Car__Aが格納されている。一方、第2階層の中間データの項目＝親図番に項目値＝Car__Aが格納されているレコードは、レコード0及びレコード1であることがわかる。よって、第1階層の中間データと第2階層の中間データとの間に、

0 → 0, 1

というレコード番号対応表が作成される。

[0074] ステップ4：レコード番号対応表の作成（2回目）

レコード番号対応表は、共通化のためのキー項目に関して項目値が一致するレコードが存在しなくなるまで続けられる。具体的には、ステップ3において作成されたレコード番号対応表に出現した第2階層の中間データのレコード0及びレコード1に関して、さらに親子関係のあるデータが存在するかどうかが調べられる。本例では、第3階層の中間データとして、原データである部品表がさらに準備される。部品表展開の場合、中間データとして原データが繰り返し利用される。なぜならば、ストラクチャー型部品表は、1つの表の中に階層関係が埋め込まれているからである。第2階層の中間データと第3階層の中間データとの間では、第2階層の中間データの項目＝子図番と第3階層の中間データの項目＝親図番が共通化のためのキー項目として選択される。

[0075] 第2階層の中間データのレコード0の項目＝子図番は、項目値＝Body__1であり、レコード1の項目＝子図番は、項目値＝Engine__1である。一方、第3階層の中間データ、すなわち、原データである部品表の項目＝親図番に項目値＝Body__1が格納されているレコードは、レコード4、レコード5及びレコード6である。また、第3階層の中間データ、すなわち、原データである部品表の項目＝親図番に項目値＝Engine__1が格納されているレコードは存在しない。よって、第2階層の中間データと第3階層の中間データとの間に、

0 → 4, 5, 6

1 →

というレコード番号対応表が作成される。

[0076] ステップ5:レコード番号対応表の作成(3回目)

第2階層と第3階層との間に作成されたレコード番号表を参照して、第3階層の中間データのレコード4、レコード5及びレコード6に属する項目＝子図番と、第4階層の中間データ、すなわち、原データである部品表の項目＝親図番とをキー項目として、さらにレコード番号対応表を作成する。しかし、本例では、第3階層の中間データのレコード4の項目値＝Tire__1、レコード5の項目値＝Chassis__1及びレコード6の項目値＝Door__1に対応する項目値が第4階層の中間データのレコードの項目＝親図番に存在しない。よって、第4階層の中間データ、及び、第3階層と第4階層との間のレコード番号対応表を作成する必要がないことがわかった。

[0077] 図9は、上記のステップ1からステップ5の処理によって、ストラクチャー型部品表から作成された中間データと中間データ間のレコードの対応関係を示す図である。中間データTable__0と中間データTable__1との間の矢印、及び、中間データTable__1と中間データTable__2との間の矢印は、それぞれ、第1階層と第2階層との間のレコード番号対応表、及び、第2階層と第3階層との間のレコード番号対応表と等価である。

[0078] ステップ6:レコード番号対応表からツリー構造データの生成

次に、取得されたレコード番号対応表からツリー構造データを生成する。上述されているように、Table__0とTable__1との間のレコード番号対応表は、

0 → 0, 1

であり、Table__1とTable__2との間のレコード番号対応表は、

0 → 4, 5, 6

1 →

である。図10A、Bには生成されたツリー構造データが示されている。図10Aはストラクチャー型部品表から生成された深さ優先形式によるツリー構造データの一例の説明図であり、図10Bは同じストラクチャー型部品表から生成された幅優先形式によるツリー構造データの一例の説明図である。

[0079] レコード番号対応表からツリー構造データを生成するステップは、深さ優先又は幅優先の何れかの方式で、レコード番号対応表から、親子関係のある親側の中間データ中のレコード番号と子側の中間データ中のレコード番号を探索するステップ6－1と、ツリー構造データの仮想的なレコードの配列中のレコードを生成するステップ6－2とを含む。

[0080] ステップ6－1において、深さ優先探索を実行すると、深さ優先形式によるツリー構造データが生成され、幅優先探索を実行すると、幅優先形式によるツリー構造データが生成される。深さ優先と幅優先の何れを選択するかは、予め設定されていてもよく、ユーザによって毎回指定されてもよく、又は、アプリケーションに応じて自動的に選択されてもよい。

[0081] たとえば、本例において、深さ優先探索を実行すると、

(1)Table__0[0] → Table__1[0]

(2) Table__1[0] → Table__2[4]

(3) Table__1[0] → Table__2[5]

(4) Table__1[0] → Table__2[6]

(5)Table__0[0] → Table__1[1]

という順番で親子関係のあるレコードが取得される。

[0082] ステップ6－2は、ステップ6－1において1つの親子関係が取得されるたびに実行される。たとえば、ステップ6－1において、最初に、親子関係(1)が取得されると、新たに仮想的なレコードの配列に作成されるべきレコードにレコード順序番号0が割り

当てられ、レコード順序番号0が割り当てられたレコードRecord[0]には、このレコードに対応するノードの親ノードのレコード順序番号Topology[0]と、このレコードに対応する原データの項目名Title[0]と、このレコードに対応する原データの項目値Value[0]がそれぞれ格納される。本例では、Record[0]は、中間データTable__0[0]に対応している。このレコードに対応するノードには親ノードが存在しないので、Topology[0]＝－1である。また、Title[0]にはTable__0[0]の項目名＝展開指令が格納され、Value[0]にはTable__0[0]の項目値＝Car__Aが格納される。続いて、親子関係(1)の子側のレコードに関する新たなレコードがツリー構造データに生成される。そのため、子側のレコードに対応して新たに生成されるべきレコードには次のレコード順序番号1が割り当てられる。よって、新たに生成されるレコードはRecord[1]である。そして、Topology[1]には、このレコードに対応するノードの親ノードに対応するレコードのレコード順序番号、すなわち、0が格納され、Title[1]には、Table__1の項目名＝子図番が格納され、Value[1]にはTable__1[0]の項目＝子図番に収容されている項目値＝Body__1が格納される。ステップ6－2において、新たなレコードが生成されると、処理はステップ6－1へ戻り、次の親子関係が探索される。

[0083] 本例では、次に、親子関係(2)が見つけられ、再びステップ6－2へ進み、新たなレコードが生成される。このステップでは、既に、親側のTable__1[0]に関するレコードは生成されているので、子側のTable__2[4]に関するレコードRecord[2]がRecord[1]と同様に生成される。レコード2に対応するノードの親ノードはレコード1であるため、Topology[2]＝1である。また、Title[2]には、Table__2の項目名＝子図番が格納され、Value[2]には、Table__2[4]の項目＝子図番に収容されている項目値＝Tire__1が格納される。以下、深さ優先方式でステップ6－1とステップ6－2を繰り返し実行することにより、図10Aに示されているようなツリー構造データを表す仮想的なレコードの配列が生成される。

[0084] 幅優先方式の場合も、深さ優先方式と同様にステップ6－1とステップ6－2を繰り返し実行することにより、図10Bに示されているようなツリー構造データを表す仮想的なレコードの配列が生成される。

[0085] ステップ7：値の分離

図4A、Bを参照して説明したように、ツリー構造データ中の項目＝Titleと項目＝Valueは、項目値配列と項目値配列へのポインタ配列とに分離することができる。よって、好ましい一実施形態では、ツリー構造データに基づく検索・集計・ソートなどの処理を効率化するため、項目＝Titleと項目＝Valueは情報ブロック型に変換される。一方、項目＝Topologyは要素が整数の配列によって記述されているので、情報ブロック型に変換する必要はない。

[0086] 図11A、Bは、図10A、Bに示されたツリー構造データを情報ブロック型へ変換することによって生成されたツリー構造データを示す図である。図11Aは図10Aの深さ優先型のツリー構造データに対応し、図11Bは図10Bの幅優先型のツリー構造データに対応している。

[0087] 以上の説明のように、本発明の一実施形態によれば、ストラクチャー型部品表から統合されたツリー構造データを生成することが可能である。ツリー構造データは、必要に応じて、深さ優先、幅優先の何れの形式でも生成できる。

[0088] ［複数のテーブルで記述されたツリー構造データの統合］

特許文献1又は特許文献2に記載では、ツリー構造データは複数のテーブルの連鎖又は連結によって表現されている。そのため、ツリー構造データを対象とする検索・ソート・集計などの操作が容易ではない。そこで、本発明の一実施形態では、複数のテーブルで記述されたツリー構造データを統合されたツリー構造データに変換する方法が提供される。

[0089] 図12A、Bは、複数のテーブルで記述されたツリー構造データの一例の説明図であり、図12Aには、ツリー構造データを記述する複数のテーブルが示され、図12Bには、複数のテーブルによって記述されているツリー構造データの階層関係が示されている。

[0090] 図12Aにおいて、1番目のテーブルTable__0は、項目Storyと項目DGripとを保有している。ここで、項目Storyは、データに属している実体的な情報であり、項目DGripはテーブルの要素間の親子関係を表現するために利用される情報である。2番目のテーブルTable__1は、項目UGripと項目Familyと項目DGripとを保有している。ここでも、項目Familyは実体的な情報であり、項目UGrip及び項目DGripは親

子関係を表現するために利用される情報である。3番目のテーブルTable__2は、親子関係を表現するための情報である項目UGripと、実体的な情報である項目Nameとを保有している。DGripとして表記されているダウングリップは、データの出現順に割り当てられた順番号である。UGripとして表記されているアップグリップは、親エレメントに割り当てられたダウングリップの値である。よって、一般に、あるテーブル中のあるレコードiのUGripの値がmであり、直前のテーブル中のレコードjのDGripの値がmであるならば、あるテーブル中のレコードiの親エレメントは、直前のテーブル中のレコードjである。それぞれのレコードは、ツリー構造データ中の1つのノードに対応し、それぞれのレコードの実体的な情報である項目名と項目値が、当該レコードに対応するノードに属する属性名及び属性値を表している。このように、アップグリップとダウングリップを組み合わせることにより、ツリー構造データが複数のテーブルで記述されている。

[0091] 図12Bを参照すると、図12Aの複数のテーブルは、Harry Potterをルートノードとするツリーと、Mr. Incredibleをルートノードとするツリーの2つのツリーを含むツリー構造データを表していることがわかる。このような複数のテーブルは、予めデータのスキャン、又は、データ構造の定義などによって、どのようなデータ構造が存在しているかを知り、それに応じて格納先テーブルを作成し、データを格納するという手順に従って構築されるので、様々なデータ構造が随時生成されるエディタのようなアプリケーションには、性能の面で不適当である。

[0092] [深さ優先方式の統合されたツリー構造データ]

最初に、複数のテーブルで記述されたツリー構造データ、すなわち、原データを深さ優先方式の統合されたツリー構造データへ変換する本発明の一実施形態を説明する。以下に述べる処理ステップは、コンピュータシステム10、特に、コンピュータシステム10中のCPU12により実行される。また、処理ステップにおいて生成されるデータ、表、配列は、それぞれ、メモリ(たとえば、RAM14)中に格納される。

[0093] ステップ1:レコード番号対応表の作成

第1階層のテーブルTable__0と第2階層のテーブルTable__1との間でレコード番号対応表が作成される。レコード番号対応表は、Table__0の項目DGripとTable__

1の項目UGripをキー項目として、Table__0とTable__1の2つのテーブルをジョインすることにより作成される。なお、本発明の一実施形態によれば、ジョインはレコード番号対応表を作成するために使用されるので、ジョインテーブルを生成する必要がないことに注意すべきである。Table__0のレコード0と、Table__1のレコード1及びレコード2のキー項目値＝0が一致し、Table__0のレコード1とTable__1のレコード0のキー項目値＝1が一致しているので、

0 → 1, 2

1 → 0

というレコード対応表が完成される。第2階層のテーブルTable__1と第3階層のテーブルTable__2との間でも同様にレコード番号対応表を作成すると、

0 → 4, 5, 6, 7

1 → 1, 2, 3

2 → 0

というレコード番号対応表が得られる。図13A、Bは、本発明の一実施形態によるレコード番号対応表の作成処理の説明図であり、図13Aには、Table__0とTable__1との間のレコード番号対応表が示され、図13Bには、Table__1とTable__2との間のレコード番号対応表が示されている。

[0094] ステップ2:レコード番号対応表からツリー構造データの生成

次に、取得されたレコード番号対応表からツリー構造データを生成する。具体的には、仮想的なレコードの配列の各レコードにTopologyフィールドの値、Titleフィールドの値、及び、Valueフィールドの値を格納する。

[0095] ステップ2－1:レコード0の生成

ステップ2－1－1:初期化

配列Topology、配列Title、配列Value、及び、親ノードのレコード識別番号を格納するスタック配列ParentNodeNoの領域をメモリに確保し、スタック配列の先頭、すなわち、ParentNodeNo[0]に－1をセットする。さらに、スタック配列ParentNodeNoに格納されている値－1をTopology[0]へコピーする。図14A－Cは、本発明の一実施形態による深さ優先方式に基づくツリー構造データ生成処理の説明図であ

り、図14Aには、ステップ2－1－1の処理が示されている。

[0096] ステップ2－1－2:値の格納

図14Bには、ルートノード(ノード0)に対応するレコード0の生成処理が示されている。同図に上向き矢印で示されているレコード番号対応表中の現在位置は、Table__0のレコード0であるので、Table__0のレコード0の項目名及び項目値を取得して、Title[0]及びValue[0]にそれぞれ格納する。

[0097] ステップ2－1－3:子ノードへの移動

図14Cには、子ノードへの移動処理が示されている。同図に示されているように、レコード番号対応表に従って、現在位置を子ノードへ移動させる。このとき、親ノードの格納アドレス(添字)、すなわち、レコード順序番号＝0をスタック配列ParentNodeNoにプッシュする。また、同図に横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0098] ステップ2－2:レコード1の生成

図15A－Cには、ノード1に対応するレコード1を生成する処理が示されている。最初に、図15Aに示されているように、スタック配列ParentNodeNoに格納されている値を配列Topology[1]へコピーする。次に、図15Bに示されているように、レコード番号対応表中の現在位置は、Table__1のレコード1であるので、Table__1のレコード1の項目名及び項目値を取得して、Title[1]及びValue[1]にそれぞれ格納する。最後に、図15Cに示されているように、レコード番号対応表に従って、現在位置を子ノードへ移動させる。このとき、親ノードの格納アドレス(添字)、すなわち、レコード順序番号＝1をスタック配列ParentNodeNoにプッシュする。また、同図に横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0099] ステップ2－3:レコード2の生成

図16A－Cには、ノード2に対応するレコード2を生成する処理が示されている。最初に、図16Aに示されているように、スタック配列ParentNodeNoに格納されている

値を配列Topology[2]へコピーする。次に、図16Bに示されているように、レコード番号対応表中の現在位置は、Table__2のレコード1であるので、Table__2のレコード1の項目名及び項目値を取得して、Title[2]及びValue[2]にそれぞれ格納する。最後に、図16Cに示されているように、レコード番号対応表に従って、子ノードが存在するかどうかをチェックし、子ノードが無ければ、弟ノードが存在するかどうかをチェックする。本例では、弟ノードが存在するので、レコード番号対応表中の現在位置を弟ノードへ移動させる。深さ優先方式であるため、子ノードが弟ノードより優先して調べられている。現在位置を弟ノードへ移動させるとき、スタック配列ParentNodeNoは操作されない。また、図16Cに横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0100] ステップ2－4:レコード3の生成

図17A－Cに示されているように、図16A－Cに示されたステップ2－3と同様にレコード3が生成される。

[0101] ステップ2－5:レコード4の生成

図18A－Cには、ノード4に対応するレコード4を生成する処理が示されている。最初に、図18Aに示されているように、スタック配列ParentNodeNoに格納されている値を配列Topology[4]へコピーする。次に、図18Bに示されているように、レコード番号対応表中の現在位置は、Table__2のレコード3であるので、Table__2のレコード3の項目名及び項目値を取得して、Title[4]及びValue[4]にそれぞれ格納する。最後に、図18Cに示されているように、レコード番号対応表に従って、子ノードが存在するかどうかをチェックし、子ノードが無ければ、弟ノードが存在するかどうかをチェックする。本例では、弟ノードも存在しないので、レコード番号対応表中の現在位置を親ノードへ移動させる(戻す)。現在位置を親ノードへ戻すとき、スタック配列ParentNodeNoから格納されているレコード順序番号をポップする。親ノードは、Table__1のレコード1に対応している。次に、この親ノードに弟ノードが存在するかどうかをチェックする。本例では、弟ノードが存在するので、レコード番号対応表中の現在位置をTable__1のレコード1の弟ノード、すなわち、Table__1のレコード2へ移動させる。

また、図18Cに横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0102] ステップ2－6:レコード5の生成

図19A－Cには、ノード5に対応するレコード5を生成する処理が示されている。最初に、図19Aに示されているように、スタック配列ParentNodeNoに格納されている値を配列Topology[5]へコピーする。次に、図19Bに示されているように、レコード番号対応表中の現在位置は、Table__1のレコード2であるので、Table__1のレコード2の項目名及び項目値を取得して、Title[5]及びValue[5]にそれぞれ格納する。最後に、図19Cに示されているように、レコード番号対応表に従って、現在位置を子ノードへ移動させる。このとき、親ノードの格納アドレス(添字)、すなわち、レコード順序番号＝5をスタック配列ParentNodeNoへプッシュする。また、図19Cに横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0103] ステップ2－7:レコード6の生成

図20A－Cには、ノード6に対応するレコード6を生成する処理が示されている。最初に、図20Aに示されているように、スタック配列ParentNodeNoに格納されている値を配列Topology[6]へコピーする。次に、図20Bに示されているように、レコード番号対応表中の現在位置は、Table__2のレコード0であるので、Table__2のレコード0の項目名及び項目値を取得して、Title[6]及びValue[6]にそれぞれ格納する。最後に、図20Cに示されているように、レコード番号対応表に従って、子ノードが存在するかどうかをチェックする。子ノードが存在しないので、弟ノードが存在するかどうかをチェックする。本例では、弟ノードも存在しないので、レコード番号対応表中の現在位置を親ノードへ移動させる(戻す)。現在位置を親ノードへ戻すとき、スタック配列ParentNodeNoから格納されているレコード順序番号をポップする。親ノードは、Table__1のレコード2に対応している。次に、この親ノードに弟ノードが存在するかどうかをチェックする。本例では、弟ノードも存在しないので、レコード番号対応表中の

現在位置をTable__1のレコード2の親ノード、すなわち、Table__0のレコード0へ移動させる。このとき、スタック配列ParentNodeNoから格納されているレコード順序番号をポップする。また、図20Cに横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0104] 以上の処理により、レコード番号対応表中の現在位置がTable__0のレコード0まで戻ったので、Harry Potterに関する深さ優先方式の統合されたツリー構造データが完成した。

[0105] もう一方のTable__0のレコード1のMr. Incredibleに関する深さ優先方式の統合されたツリー構造データも同様の処理によって生成される。Table__0のレコード1の処理は、1台のプロセッサによって、Table__0のレコード0に関する処理の後に逐次的に実行されてもよいが、たとえば、複数台のプロセッサを用いて、Table__0のレコード0に関する処理と、Table__0のレコード1に関する処理が並列的に実行されてもよい。

[0106] ステップ3:値リストが分離されたツリー構造データの生成

ステップ2で生成されたツリー構造データは、値リストが分離されたツリー構造データに変換することが可能である。そして、値リストが分離されたツリー構造データは、情報ブロック型のツリー構造データと等価である。よって、本発明の一実施形態では、統合されたツリー構造データを値リストが分離されたツリー構造データに変換する方法が提供される。図21A－Cは、図12Aの原データから生成された、値リストが分離されたツリー構造データの説明図である。図21Aには統合されたツリー構造データが示されている。

[0107] 本実施形態では、配列Titleに出現する値が値番号と一意かつ昇順(又は降順)の値リストとに分離され、配列Valueに出現する値もまた値番号と一意かつ昇順(又は降順)の値リストとに分離される。たとえば、配列Titleを参照すると、配列Title中に出現する値は、

Story

Name

Family

の3個である。これらの3個の値をアルファベット順に並べ替えると、

Family

Name

Story

の順になる。よって、値Familyに値番号0を、値Nameに値番号1を、値Storyに値番号2を割り当て、かつ、

VL[0]＝Family

VL[1]＝Name

VL[2]＝Story

という値リストを準備すると、値は値番号VNoと値リストVLに分離される。図21Bには、このようにして、値が値番号と値リストに分離されたツリー構造データが示されている。図21Cには、値番号と値リストの組を項目Titleと項目Valueに関して分離した情報ブロック型のツリー構造データが示されている。

[0108] このように値リストが分離されると、配列Titleや配列Valueが整数型配列になるので、コンピュータで高速に処理できるようになる。また、値リストの分離によって重複値の保持が無くなるので、データサイズがコンパクトになる。さらに、値リストの分離によって、検索の際に、同じ値について検索条件の成立を何回も調べる必要がなくなるので、検索が容易になる。また、値リストの分離によって、集計の際に、次元値の一覧が即座に利用できるので、集計が容易になる。その上、値リストの分離によって、ソートの際に、カウンティングソートを利用できるので、ソートが容易になる。

[0109] [順序集合と値リストを用いたツリー構造データの生成]

上記の本発明の一実施形態では、ツリー構造データ中の値は、統合されたツリー構造データが作成された後に、値番号と値リストに分離されている。これに対して、本発明の他の実施形態では、複数のテーブルで記述されたツリー構造データ、すなわち、原データを順序集合と情報ブロックとに分離した後に、統合されたツリー構造データを生成する方法が提供される。以下に述べる処理ステップも、コンピュータシステム10、特に、コンピュータシステム10中のCPU12により実行される。また、処理ステ

ップにおいて生成されるデータ、表、配列は、それぞれ、メモリ(たとえば、RAM14)中に格納される。

[0110] ステップ1:データ形式の変換

最初に、図12Aに示された原データ、すなわち、Table__0、Table__1及びTable__2は、順序集合配列OrdSetと情報ブロックとからなるデータ形式に変換される。このデータ形式は、既に[情報ブロックに基づくデータ管理機構]において説明した通りである。図22は、図12Aに示された原データから情報ブロックに基づくデータ形式への変換の説明図である。

[0111] ステップ2:項目Valueのための値リストの共通化

次に、項目Story、項目Family及び項目Nameの値リストが共通化される。そのため、最初に項目Storyの値リストと項目Familyの値リストを共通化し、その後に、最初に共通化された値リストと項目Nameの値リストを共通化する。値リストの共通化は、最終的な情報ブロック型のツリー構造データの項目Valueの情報ブロックを作成するために必要な処理である。図23は、値リストの共通化処理の説明図である。項目Storyの値リストと項目Familyの値リストを共通化すると、共通化された値リストには、それぞれの値リストからの要素が、重複することなく、アルファベット順に格納されている。そして、各項目の値番号配列の要素は、共通化された値リスト中の対応する値を指定するように値番号が書き換えられる。複数の項目の値リストを共通化すると、複数の項目を1つの情報ブロックとして取り扱えるようになる。

[0112] ステップ3:項目Titleの値番号と値リストの分離

次に、ツリー構造データの項目Titleを記述するため、Titleのリストを値番号配列VNoと値リストVLに分離する。図24は、値配列を値番号と値リストへ分離する処理の説明図である。

[0113] ステップ4:レコード番号対応表の作成

次に、Table__0とTable__1との間、及び、Table__1とTable__2との間でレコード番号対応表が作成される。この処理は図13を参照して説明した処理と同様である。この処理は、上述されているように、特許文献5に記載されたジョイン処理を利用することによって実現される。図25A、Bは、本発明の一実施形態による、ジョインを利用

するレコード番号対応表の作成処理の説明図である。図25AはTable__0とTable__1との間のジョイン処理を示し、図25BはTable__1とTable__2との間のジョイン処理を示している。図25A、Bにおいて、マスタ側とは、元のテーブルの順序集合の順序が保存されている方のテーブルを表し、本例では、上位階層のテーブルがマスタ側に相当し、反対に、スレイブ側とは、元のテーブルの順序集合の順序がジョインキーでソートされている方のテーブルを表し、本例では、下位階層のテーブルがスレイブ側に相当する。

[0114] 図25Bを参照して、ジョインを利用した、レコード番号対応表の作成処理を詳細に説明する。Table__1とTable__2をジョインするとき、キー項目は、マスタ側であるTable__1のDGripと、スレイブ側であるTable__2のUGripである。たとえば、マスタ側テーブルのレコードmのキー項目値と一致するキー項目値を有する要素がスレイブ側データにn個存在している場合、ジョインによって生成されるジョインテーブルには、マスタ側テーブルのレコードmがn回繰り返して出現する。本例では、マスタ側のTable__1のレコード0に対応するスレイブ側のTable__2のレコードは、レコード4、レコード5、レコード6及びレコード7の4個存在している。同様に、マスタ側のレコード1には、スレイブ側のレコード1、レコード2及びレコード3が対応し、マスタ側のレコード2には、スレイブ側のレコード0が対応している。図25Bに示されているマスタ側累計数配列と、マスタ側順序集合配列と、マスタ側ジョインキー配列と、スレイブ側累計数配列と、スレイブ側順序集合配列は、このようなマスタ側レコードとスレイブ側レコードの対応関係を記述している。

[0115] マスタ側累計数配列は、スレイブ側の対応するレコードの個数を累計数化することにより得られる配列であり、マスタ側累計数配列[1]＝4から、マスタ側順序集合配列[0]＝0によって指定されるマスタ側レコード0に対応するスレイブ側レコードの個数が4個であることがわかる。同様に、マスタ側累計数配列[2]＝7から、マスタ側順序集合配列[1]＝1によって指定されるマスタ側レコード1に対応するスレイブ側レコードの個数が7－4＝3個であることがわかる。さらに、マスタ側累計数配列[3]＝8から、マスタ側順序集合配列[2]＝2によって指定されるマスタ側レコード2に対応するスレイブ側レコードの個数が8－7＝1個であることがわかる。よって、マスタ側累計数配

列とマスタ側順序集合配列の組み合わせは、図26Aに示されているような仮想的な順序集合配列をコンパクトに表現していることがわかる。換言すると、マスタ側累計数配列[i]は、対応するマスタ側順序集合配列[i]で指定される要素が仮想的な順序集合配列中に出現する開始位置を表している。

[0116] スレイブ側順序集合は、ジョインキー、本例では、UGripの値でソートされている。図26Bに示されているように、スレイブ側累計数配列[i]もまた、共通化されたジョインキーVL[i]に対応する要素がスレイブ側順序集合配列中に出現する開始位置を示している。

[0117] 図25Bに示されているようなデータを取得する処理の一例として、特許文献5に記載されているジョイン処理を簡単に説明する。

[0118] 最初に、複数の表形式データの間で、共通化すべき項目(ジョインキー)を見出す。次に、複数の表形式データにおいて、値リストを比較して、双方の値リストを等価にする。値リストを等価にする際には、項目値の追加に伴ってポインタ配列を変換して、新たなポインタ配列を生成する。さらに、スレイブ表形式データに関するレコードの個数を示す存在数を、項目値に対応して格納するスレイブ側存在数配列を生成し、スレイブ側存在数配列中の存在数を参照して、項目値に対応した存在数の累計数を格納したスレイブ側累計数配列を生成する。一方、マスタ側のレコード番号が示すポインタ配列中のポインタ値を取り出し、ポインタ値が指し示す項目値に対応したスレイブ側存在数配列中の要素を特定し、マスタ側のレコード番号と対応付けて、マスタ表形式データの各レコードに対応するスレイブ表形式データのレコード数を示すレコード数指示配列中に収容し、レコード数指示配列中のレコード数を参照して、マスタ側のレコード番号に対応した、レコード数の累計数を格納するマスタ側累計数配列を生成する。さらに、マスタ側累計数配列に関して、レコード数の累計数の総和を求め、当該総和の数だけの要素を収容可能な、結合された表形式データに関する新たなレコードを特定するための新たなレコード番号配列中の新たなレコード番号と、マスタ側累計数配列中の要素とを比較することにより、重複を考慮した、マスタ表形式データにおけるレコード番号を収容したマスタ側順序集合配列を求める。次に、マスタ側順序集合配列中の要素であるマスタ表形式データにおけるレコード番号により指し示さ

れる、スレイブ表形式データに関するポインタ配列中の要素を特定し、スレイブ表形式データに関するポインタ配列中の要素が指し示す、スレイブ側累計数配列中の要素を特定し、これをスレイブ側の開始アドレスとして一時的に保持し、新たなレコード番号配列中のレコード番号と、当該レコード番号により特定されるマスタ累計数配列中の要素と、スレイブ側の開始アドレスとから、重複を考慮した、スレイブ表形式データにおけるレコード番号を収容したスレイブ側順序集合配列を求める。

[0119] もう一度図25Bに戻り、マスタ側レコードから対応するスレイブ側レコードを取得する処理を説明する。マスタ側順序集合配列[0]＝0と、対応するマスタ側ジョインキーVNo「0」＝0とから、マスタ側レコード0に対応するスレイブ側レコードのレコード番号は、スレイブ側順序集合配列中のスレイブ側累計数配列[0]＝0番目から、スレイブ側累計数配列[0＋1]＝4－1＝3番目までの要素、すなわち、レコード番号＝4、5、6及び7であることがわかる。これにより、レコード番号対応表の1行目の

0 → 4, 5, 6, 7

が取得できる。同様に、マスタ側レコード1に対応するスレイブ側レコードは、スレイブ側順序集合配列中のスレイブ側累計数配列[1]＝4番目から、スレイブ側累計数配列[1＋1]＝7－1＝6番目までの要素、すなわち、レコード番号＝1、2及び3であることがわかる。これにより、レコード番号対応表の2行目の

1 → 1, 2, 3

が取得できる。さらに、同様に、マスタ側レコード2に対応するスレイブ側レコードは、スレイブ側順序集合配列中のスレイブ側累計数配列[2]＝7番目から、スレイブ側累計数配列[2＋1]＝8－1＝7番目までの要素、すなわち、レコード番号＝0であることがわかる。これにより、レコード番号対応表の3行目の

2 → 0

が取得できる。

[0120] Table__0とTable__1との間のレコード番号対応表もジョイン処理を利用して同様に取得できる。

[0121] ステップ5:ツリー構造データの生成

ステップ4において取得されたレコード番号対応表が生成されると、ツリー構造デー

タのTopology配列と、Title__VNo配列と、Value__VNo配列に、レコード0から順に値を格納することにより、値が分離されたツリー構造データが生成される。このとき、図23に示されているような、項目Story、項目Family及び項目Nameを共通化した項目Valueに関する値番号及び値リストと、図24に示されているような、項目Titleに関する値番号及び値リストが使用される。

[0122] ステップ5－1:レコード0の生成

ステップ5－1－1:初期化

配列Topology、配列Title、配列Value、及び、親ノードのレコード識別番号を格納するスタック配列ParentNodeNoの領域をメモリに確保し、スタック配列の先頭、すなわち、ParentNodeNo[0]に－1をセットする。さらに、スタック配列ParentNodeNoに格納されている値－1をTopology[0]へコピーする。この処理は、図14Aを参照して説明した処理に類似している。

[0123] ステップ5－1－2:値の格納

この処理は、図14Bを参照して説明したルートノード(ノード0)に対応するレコード0の生成処理と類似している。レコード番号対応表中の現在位置は、Table__0のレコード0であるので、Table__0のレコード0の項目名storyの値番号＝2、及び、項目値の値番号4を取得して、Title[0]及びValue[0]にそれぞれ格納する。このように、本実施形態では、配列Title及び配列Valueに格納される値が、項目名及び項目値ではなく、項目名の値番号及び項目値の値番号である。

[0124] ステップ5－1－3:子ノードへの移動

この処理は、図14Cを参照して説明された子ノードへの移動処理と類似している。レコード番号対応表に従って、現在位置を子ノードへ移動させる。このとき、親ノードの格納アドレス(添字)、すなわち、レコード順序番号＝0をスタック配列ParentNodeNoにプッシュする。また、同図に横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0125] ステップ5－2:レコード1の生成

この処理は、図15A－Cを参照して説明されたノード1に対応するレコード1を生成

する処理と類似している。最初に、スタック配列ParentNodeNoに格納されている値を配列Topology[1]へコピーする。次に、レコード番号対応表中の現在位置は、Table__1のレコード1であるので、Table__1のレコード1の項目名Familyの値番号＝0及び項目値Potterの値番号＝10を取得して、Title[1]及びValue[1]にそれぞれ格納する。最後に、レコード番号対応表に従って、現在位置を子ノードへ移動させる。このとき、親ノードの格納アドレス(添字)、すなわち、レコード順序番号＝1をスタック配列ParentNodeNoにプッシュする。また、図15Cに横向き矢印で示されている配列Topology、配列Title、及び、配列Valueへのポインタが値を格納済みのレコード位置を指しているので、当該ポインタをインクリメントする(1つ先へ進める)。

[0126] 以下、図16A－C乃至図20A－Cを参照して説明した処理と同様の処理を実行することにより、図21Bに示されているように、値リストが分離されたツリー構造データが生成される。この値リストが分離されたツリー構造データを、図21Cに示されるような情報ブロック型のツリー構造データへ変換する処理は前述されている通りである。

[0127] 最後に、もう一方のTable__0のレコード1のMr. Incredibleに関する深さ優先方式のツリー構造データも同様の処理によって生成される。Table__0のレコード1の処理は、1台のプロセッサによって、Table__0のレコード0に関する処理の後に逐次的に実行されてもよいが、たとえば、複数台のプロセッサを用いて、Table__0のレコード0に関する処理と、Table__0のレコード1に関する処理が並列的に実行されてもよい。

[0128] 図27には、Table__0のレコード0とレコード1の両方に対して、本発明の一実施形態によるツリー構造データの生成処理を適用した結果が示されている。この例では、Table__0のレコード0が一方のルートノードに対応し、Table__1のレコード1がもう一方のルートノードに対応している。これに対して、Table__0のレコード0に対応するノードとTable__1のレコード1に対応するノードが兄弟ノードとなるようなツリー構造データを生成することも可能である。図28には、Table__0のレコード0とレコード1を兄弟ノードとして取り扱うような、本発明の別の実施形態によるツリー構造データの生成処理を適用した結果が示されている。

[0129] [幅優先方式の統合されたツリー構造データ]

今度は、複数のテーブルで記述されたツリー構造データ、すなわち、原データを幅優先方式の統合されたツリー構造データへ変換する本発明の一実施形態を説明する。ここでも、原データは、図12Aに示されているような複数のテーブルで記述されたツリー構造データである。以下に述べる処理ステップも、コンピュータシステム10、特に、コンピュータシステム10中のCPU12により実行される。また、処理ステップにおいて生成されるデータ、表、配列は、それぞれ、メモリ(たとえば、RAM14)中に格納される。

[0130] ステップ1:レコード番号対応表の作成

図13A、Bを参照して説明した、深さ優先方式におけるレコード番号対応表の作成処理が幅優先方式においても全く同様に適用される。

[0131] ステップ2:レコード番号対応表からツリー構造データの生成

次に、取得されたレコード番号対応表からツリー構造データを生成する。具体的には、幅優先方式であるため、最上位階層から順に仮想的なレコードの配列での格納位置を決定し、仮想的なレコードの配列の各レコードにTopologyフィールドの値、Titleフィールドの値、及び、Valueフィールドの値を格納する。

[0132] ステップ3:レコード0の生成

最上位ノードは、Table__0のレコード0である。最上位ノードには親ノードが存在しないので、−1をTopology[0]に格納する(転送1)。Table__0のレコード0の項目名及び項目値を取得して、Title[0]及びValue[0]にそれぞれ格納する(転送2)。確定したレコード順序番号(すなわち、ノード番号、この例では、0)をレコード番号対応表に追記する(転送3)。ノード0には弟ノードが存在しないので、次のノードは、次の階層(深さ)に存在する最初のノードである。図29は、このような本発明の一実施形態による幅優先方式のツリー構造データ生成処理の説明図である。

[0133] ステップ4:レコード1の生成

次に、図30を参照して、ツリー構造データのレコード1の生成処理を説明する。現在のノードは、Table__1のレコード1である。親ノードのレコード順序番号は0であるので、Topology[1]に0を格納する(転送1)。Table__1のレコード1の項目名及び項目値を取得して、Title[1]及びValue[1]にそれぞれ格納する(転送2)。確定し

たノード番号(この例では、1)をレコード番号対応表に追記する(転送3)。ノード1には弟ノードが存在するので、次のノードは弟ノードである。

[0134] ステップ5:レコード2の生成

次に、図31を参照して、ツリー構造データのレコード2の生成処理を説明する。現在のノードは、Table__1のレコード2である。親ノードのレコード順序番号は0であるので、Topology[2]に0を格納する(転送1)。Table__1のレコード2の項目名及び項目値を取得して、Title[2]及びValue[2]にそれぞれ格納する(転送2)。確定したノード番号(この例では、2)をレコード番号対応表に追記する(転送3)。ノード2には弟ノードが存在せず、かつ、同世代にも未確定ノードが存在しないので、次のノードは、次の階層(深さ)に存在する最初のノードである。

[0135] ステップ6:レコード3の生成

次に、図32を参照して、ツリー構造データのレコード3の生成処理を説明する。現在のノードは、Table__2のレコード1である。親ノードのレコード順序番号は1であるので、Topology[3]に1を格納する(転送1)。Table__2のレコード1の項目名及び項目値を取得して、Title[3]及びValue[3]にそれぞれ格納する(転送2)。確定したノード番号(この例では、3)をレコード番号対応表に追記する(転送3)。ノード3には弟ノードが存在するので、次のノードは弟ノードである。

[0136] ステップ7:レコード4の生成

次に、ツリー構造データのレコード4の生成処理を説明する。現在のノードは、Table__2のレコード2である。親ノードのレコード順序番号は1であるので、Topology[4]に1を格納する(転送1)。Table__2のレコード2の項目名及び項目値を取得して、Title[4]及びValue[4]にそれぞれ格納する(転送2)。確定したノード番号(この例では、4)をレコード番号対応表に追記する(転送3)。ノード4には弟ノードが存在するので、次のノードは弟ノードである。

[0137] ステップ8:レコード5の生成

次に、図33を参照して、ツリー構造データのレコード5の生成処理を説明する。現在のノードは、Table__2のレコード3である。親ノードのレコード順序番号は1であるので、Topology[5]に1を格納する(転送1)。Table__2のレコード3の項目名及び

項目値を取得して、Title[5]及びValue[5]にそれぞれ格納する(転送2)。確定したノード番号(この例では、5)をレコード番号対応表に追記する(転送3)。ノード5には弟ノードは存在しないが、同世代の未確定ノードが存在するので、次のノードは同世代の次のデータである。

[0138] ステップ9:レコード6の生成

次に、図34を参照して、ツリー構造データのレコード6の生成処理を説明する。現在のノードは、Table__2のレコード0である。親ノードのレコード順序番号は2であるので、Topology[6]に2を格納する(転送1)。Table__2のレコード0の項目名及び項目値を取得して、Title[6]及びValue[6]にそれぞれ格納する(転送2)。確定したノード番号(この例では、6)をレコード番号対応表に追記する(転送3)。ノード6には、弟ノードも次の世代のノードも存在しないので、ツリー構造データの生成処理が完了する。

[0139] 幅優先方式によるツリー構造データの生成処理の場合も、深さ優先方式と同様に、値リストが分離されたツリー構造データの生成処理や、順序集合と値リストを用いたツリー構造データの生成処理を実現することが可能である。図35は、本発明の一実施形態による幅優先方式のツリー構造データ生成処理を用いて生成された情報ブロック型ツリー構造データの説明図である。

[0140] 最後に、もう一方のTable__0のレコード1のMr. Incredibleに関する幅優先方式のツリー構造データも同様の処理によって生成される。Table__0のレコード1の処理は、1台のプロセッサによって、Table__0のレコード0に関する処理の後に逐次的に実行されてもよいが、たとえば、複数台のプロセッサを用いて、Table__0のレコード0に関する処理と、Table__0のレコード1に関する処理が並列的に実行されてもよい。

[0141] 図36には、Table__0のレコード0とレコード1の両方に対して、本発明の一実施形態によるツリー構造データの生成処理を適用した結果が示されている。この例では、Table__0のレコード0が一方のルートノードに対応し、Table__1のレコード1がもう一方のルートノードに対応している。これに対して、Table__0のレコード0に対応するノードとTable__1のレコード1に対応するノードが兄弟ノードとなるようなツリー構造データを生成することも可能である。図37には、Table__0のレコード0とレコード1を兄

弟ノードとして取り扱うような、本発明の別の実施形態によるツリー構造データの生成処理を適用した結果が示されている。

[0142] 本発明は、以上の実施の形態に限定されることなく、特許請求の範囲に記載された発明の範囲内で、種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることは言うまでもない。

請求の範囲

[1] 各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを各レコードがレコードの並び順を表すレコード順序番号によって識別される仮想的なレコードの配列として記憶装置上に構築する装置であって、

階層構造を有する複数のテーブルを階層毎にマッチングキーによってマッチングさせ、親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成する手段と、

前記行番号対応表から親子関係がある前記親側のテーブル中の行番号及び前記子側のテーブル中の行番号を探索する手段と、

前記探索する手段によって見つけられた順に、前記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、前記仮想的なレコードの配列中の前記レコード順序番号によって識別される位置に、親子関係フィールド値として前記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、前記ノードの属性名フィールド値として前記子側のテーブル中の行に属する項目名を格納し、前記ノードの属性値フィールド値として前記子側のテーブル中の行に属する項目値を格納する手段と、

を備える装置。

[2] 前記ノードの属性名フィールド値が前記レコード順序番号の順番に格納されている配列から、前記レコード順序番号の順に前記属性名を一意に特定する属性名番号が格納されている属性名番号配列、前記属性名番号配列の要素の順番に前記属性名番号によって一意に特定される前記属性名が格納されている属性名配列、前記レコード順序番号の順に前記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、前記属性値番号配列の要素の順番に前記属性値番号によって一意に特定される前記属性値が格納されている属性値配列を作成する手段をさらに備える、請求項1に記載の装置。

[3] 前記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、

前記行番号対応表を作成する手段が、最上位階層のテーブルとして、前記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のデータとして同

一の前記ストラクチャー型部品表を利用する、

請求項1又は2に記載の装置。

[4] 前記行番号対応表を作成する手段が、前記マッチングキーとして、前記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、前記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する、請求項1又は2に記載の装置。

[5] 階層構造を有する複数のテーブルを記憶するメモリを備え、前記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し前記メモリに格納する装置であって、

前記複数のテーブルを階層毎にマッチングキーによってマッチングさせるマッチング部と、

前記複数のテーブルの階層毎にマッチングした親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成する対応表作成部と、

前記行番号対応表から親子関係がある前記親側のテーブル中の行番号及び前記子側のテーブル中の行番号を探索する探索部と、

前記探索部によって見つけられた順に、前記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、前記仮想的なレコードの配列中の前記レコード順序番号によって識別される位置に、親子関係フィールド値として前記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、前記ノードの属性名フィールド値として前記子側のテーブル中の行に属する項目名を格納し、前記ノードの属性値フィールド値として前記子側のテーブル中の行に属する前記原データに属する項目値を格納するレコード生成部と、

を備え、

前記ツリー構造データが、各レコードがレコードの並び順を表す前記レコード順序番号によって識別される、仮想的なレコードの配列として前記メモリに構築される装置。

[6] 前記ノードの属性名フィールド値が前記レコード順序番号の順番に格納されている配列から、前記レコード順序番号の順に前記属性名を一意に特定する属性名番号

が格納されている属性名番号配列、前記属性名番号配列の要素の順番に前記属性名番号によって一意に特定される前記属性名が格納されている属性名配列、前記レコード順序番号の順に前記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、前記属性値番号配列の要素の順番に前記属性値番号によって一意に特定される前記属性値が格納されている属性値配列を作成する値分離部をさらに備える、請求項5に記載の装置。

[7] 前記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、

前記対応表作成部が、最上位階層の表構造の中間データとして、前記ツリー構造データのルートノードに対応する部品を選択し、その他の表構造の中間データとして同一の前記ストラクチャー型部品表を利用する、

請求項5又は6に記載の装置。

[8] 前記対応表作成部が、前記マッチングキーとして、前記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、前記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する、請求項5又は6に記載の装置。

[9] 各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを各レコードがレコードの並び順を表すレコード順序番号によって識別される仮想的なレコードの配列として記憶装置上に構築する方法であって、

階層構造を有する複数のテーブルを階層毎にマッチングキーによってマッチングさせ、親側のテーブルと子側のテーブルとの間で親子関係がある行番号同士の行番号対応表を作成するステップと、

前記行番号対応表から親子関係がある前記親側のテーブル中の行番号及び前記子側のテーブル中の行番号を探索し、探索によって見つけられた順に、前記子側のテーブル中の行に割り当てられるレコード順序番号を決定し、前記仮想的なレコードの配列中の前記レコード順序番号によって識別される位置に、親子関係フィールド値として前記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、前記ノードの属性名フィールド値として前記子側のテーブル中の行に属する項目名を格納し、前記ノードの属性値フィールド値として前記子側のテーブル中の

行に属する項目値を格納するステップと、

を備える方法。

[10] 前記ノードの属性名フィールド値が前記レコード順序番号の順番に格納されている配列から、前記レコード順序番号の順に前記属性名を一意に特定する属性名番号が格納されている属性名番号配列、前記属性名番号配列の要素の順番に前記属性名番号によって一意に特定される前記属性名が格納されている属性名配列、前記レコード順序番号の順に前記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、前記属性値番号配列の要素の順番に前記属性値番号によって一意に特定される前記属性値が格納されている属性値配列を作成するステップをさらに備える、請求項9に記載の方法。

[11] 前記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、

前記行番号対応表を作成するステップにおいて、最上位階層のテーブルとして、前記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のテーブルとして同一の前記ストラクチャー型部品表を利用する、

請求項9又は10に記載の方法。

[12] 前記行番号対応表を作成するステップにおいて、前記マッチングキーとして、前記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、前記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する、請求項9又は10に記載の方法。

[13] 階層構造を有する複数のテーブルを記憶するメモリを備え、前記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し前記メモリに格納するコンピュータに、

前記複数のテーブルを階層毎にマッチングキーによってマッチングさせる機能と、

前記複数のテーブルの階層毎に、マッチングした親側のテーブルと子側のテーブルとの間で、親子関係がある行番号同士の行番号対応表を作成する機能と、

前記行番号対応表から親子関係がある前記親側のテーブル中の行番号及び前記子側のテーブル中の行番号を探索する機能と、

探索によって見つけられた順に、前記子側のテーブル中の行に割り当てられる前

記レコード順序番号を決定する機能と、

前記ツリー構造データが、各レコードがレコードの並び順を表す前記レコード順序番号によって識別される、仮想的なレコードの配列として前記メモリに構築されるように、前記仮想的なレコードの配列中の前記レコード順序番号によって識別される位置に、親子関係フィールド値として前記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、前記ノードの属性名フィールド値として前記子側のテーブル中の行に属する項目名を格納し、前記ノードの属性値フィールド値として前記子側のテーブル中の行に属する項目値を格納する機能と、

を実現させるためのプログラム。

[14] 前記ノードの属性名フィールド値が前記レコード順序番号の順番に格納されている配列から、前記レコード順序番号の順に前記属性名を一意に特定する属性名番号が格納されている属性名番号配列、前記属性名番号配列の要素の順番に前記属性名番号によって一意に特定される前記属性名が格納されている属性名配列、前記レコード順序番号の順に前記属性値を一意に特定する属性値番号が格納されている属性値番号配列、及び、前記属性値番号配列の要素の順番に前記属性値番号によって一意に特定される前記属性値が格納されている属性値配列を作成する機能を前記コンピュータさらに実現させるための請求項13に記載のプログラム。

[15] 前記複数のテーブルが部品の親子関係を表現するストラクチャー型部品表であり、

最上位階層のテーブルとして、前記ツリー構造データのルートノードに対応する部品を選択し、その他の階層のテーブルとして同一の前記ストラクチャー型部品表を利用する機能を前記コンピュータにさらに実現させるための請求項13又は14に記載のプログラム。

[16] 前記マッチングキーとして、前記親側のテーブルにおけるデータの出現順に割り当てられた順序番号と、前記子側のテーブルにおけるデータに対応する親側のデータに割り当てられた順序番号との対を利用する機能を前記コンピュータにさらに実現させるための請求項13又は14に記載のプログラム。

[17] 請求項13乃至16のうちの何れか1項に記載のプログラムを記録したコンピュータ読み取り可能な記録媒体。

[18] 階層構造を有する複数のテーブルを記憶するメモリを備え、前記複数のテーブルを各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データに変換し前記メモリに格納するコンピュータに、

前記複数のテーブルを階層毎にマッチングキーによってマッチングさせる機能と、

前記複数のテーブルの階層毎に、マッチングした親側のテーブルと子側のテーブルとの間で、親子関係がある行番号同士の行番号対応表を作成する機能と、

前記行番号対応表から親子関係がある前記親側のテーブル中の行番号及び前記子側のテーブル中の行番号を探索する機能と、

探索によって見つけられた順に、前記子側のテーブル中の行に割り当てられるレコード順序番号を決定する機能と、

前記ツリー構造データが、各レコードがレコードの並び順を表す前記レコード順序番号によって識別される、仮想的なレコードの配列として前記メモリに構築されるように、前記仮想的なレコードの配列中の前記レコード順序番号によって識別される位置に、親子関係フィールド値として前記子側のテーブル中の行に対応するノードの親ノードのレコード順序番号を格納し、前記ノードの属性名フィールド値として前記子側のテーブル中の行に属する項目名を格納し、前記ノードの属性値フィールド値として前記子側のテーブル中の行に属する項目値を格納する機能と、

を実現させるためのプログラムプロダクト。

[19] 各ノードが属性名及び属性値によって表現される複数のノードからなるツリー構造データを記録したコンピュータ読み取り可能な記録媒体であって、

前記ツリー構造データは、各レコードがレコードの並び順を表すレコード順序番号によって識別される、仮想的なレコードの配列によって表現され、

レコードは、前記レコードに対応するノードの親ノードのレコード順序番号を格納する親子関係フィールドと、前記ノードに属する属性名を指定する属性名番号を格納する属性名フィールドと、前記ノードに属する属性値を指定する属性値番号を格納する属性値フィールドとを含み、

前記ノードの親ノードが前記ノードに対応する前記レコードに含まれる前記親子関係フィールドに格納されている前記レコード順序番号によって特定され、

前記ノードに属する属性名が前記ノードに対応する前記レコードに含まれる前記属性名番号と前記属性名が前記属性名番号の順番に格納されている属性名配列とによって特定され、

前記ノードに属する属性値が前記ノードに対応する前記レコードに含まれる前記属性値番号と前記属性値が前記属性名番号の順番に格納されている属性値配列とによって特定される、

コンピュータ読み取り可能な記録媒体。

[20] 前記レコード順序番号は、深さ優先探索又は幅優先探索の順で前記複数のノードのうちの各ノードに対応する前記レコードに割り当てられている、請求項19に記載のコンピュータ読み取り可能な記録媒体。

[図1]

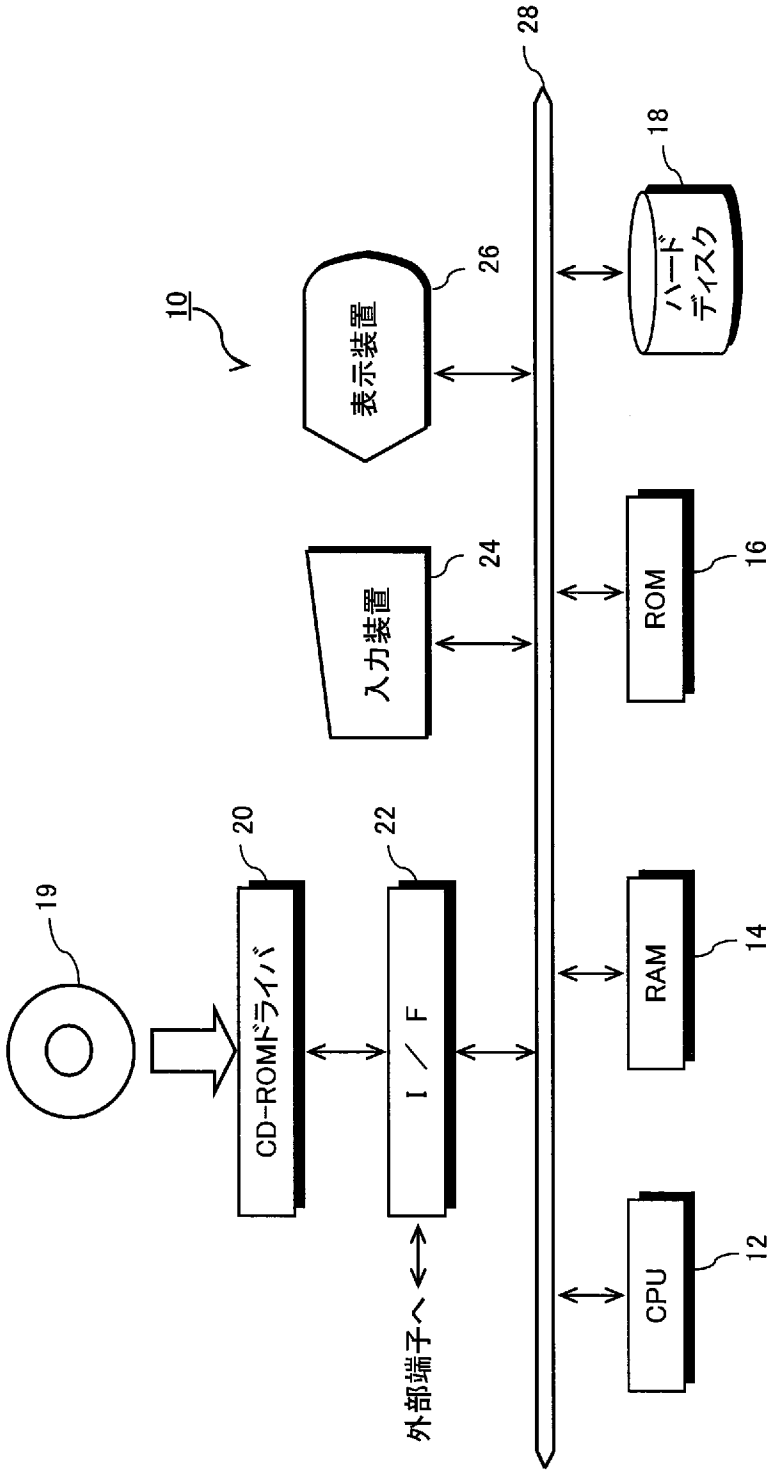
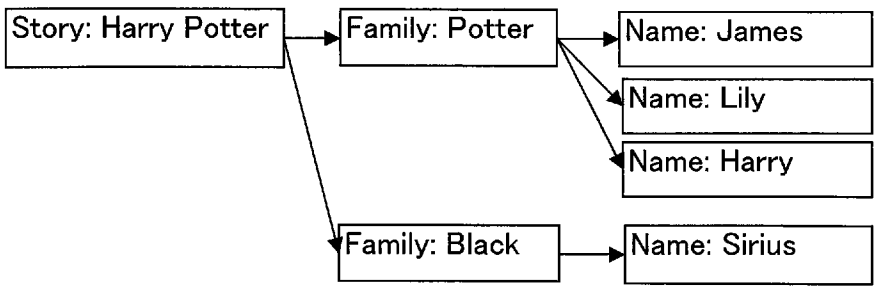

10
19
20
CD-ROMドライバ
22
I／F
外部端子へ
28
表示装置
26
入力装置
24
ハードディスク
18
ROM
16
RAM
14
CPU
12

図1

[図2]

図2



[図3]

図3A

	Topology	Title	Value
0	-1	Story	Harry Potter
1	0	Family	Potter
2	1	Name	James
3	1	Name	Lily
4	1	Name	Harry
5	0	Family	Black
6	5	Name	Sirius

図3B

	Topology	Title	Value
0	-1	Story	Harry Potter
1	0	Family	Potter
2	0	Family	Black
3	1	Name	James
4	1	Name	Lily
5	1	Name	Harry
6	2	Name	Sirius

[図4]

図4A

	Topology	Title VNo	Value VNo
0	-1	2	2
1	0	0	5
2	1	1	3
3	1	1	4
4	1	1	1
5	0	0	0
6	5	1	6

	Title VL
0	Family
1	Name
2	Story

	Value VL
0	Black
1	Harry
2	Harry Potter
3	James
4	Lily
5	Potter
6	Sirius

図4B

Topology

0	-1
1	0
2	1
3	1
4	1
5	0
6	5

Title

Title VNo	Title VL	
2	0	Family
0	1	Name
1	2	Story
1		
1		
0		
1		

Value

Value VNo	Value VL	
2	0	Black
5	1	Harry
3	2	Harry Potter
4	3	James
1	4	Lily
0	5	Potter
6	6	Sirius

[図5]

図5

レコード番号	性別	年齢	身長(cm)
0	女	3	78
1	男	1	82
2	女	2	69
3	女	1	82
4	男	3	91
5	女	1	76
6	女	1	78
7	女	2	84
8	男	3	87
9	女	3	80

[図6]

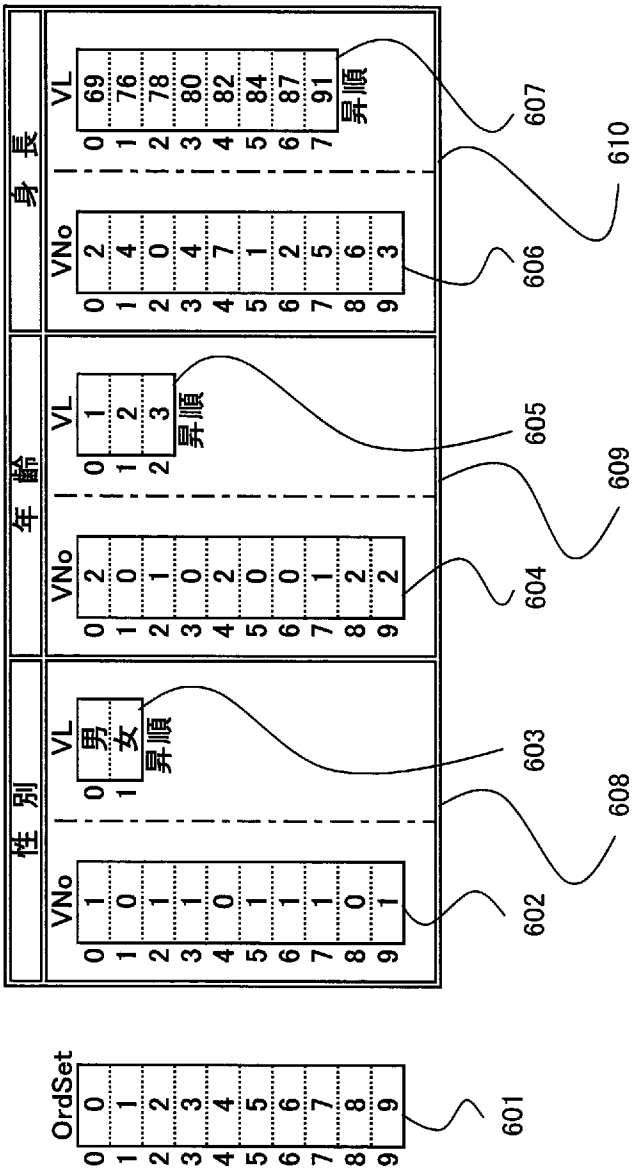


図6

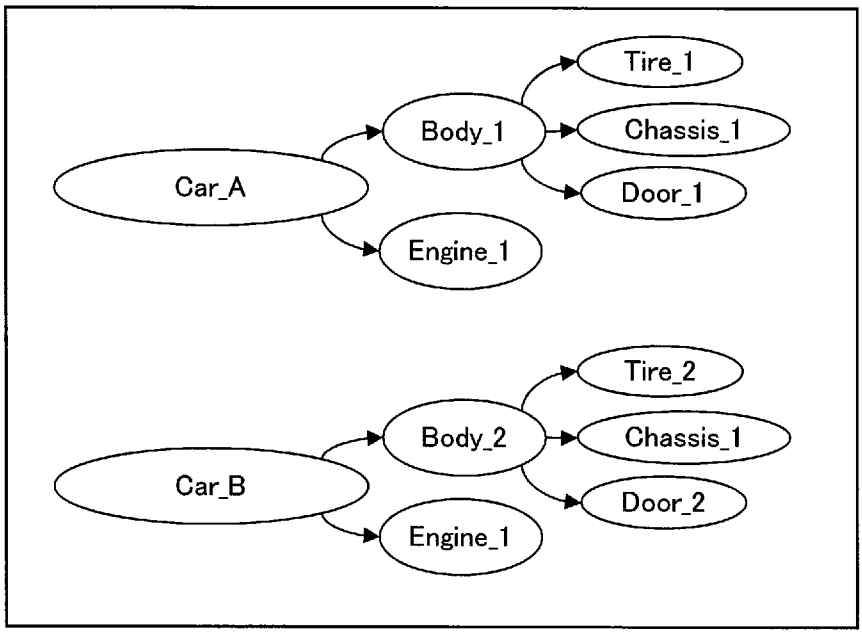
[図7]

図7

	親図番	子図番
0	Car_A	Body_1
1	Car_A	Engine_1
2	Car_B	Body_2
3	Car_B	Engine_1
4	Body_1	Tire_1
5	Body_1	Chassis_1
6	Body_1	Door_1
7	Body_2	Tire_2
8	Body_2	Chassis_1
9	Body_2	Door_2

[図8]

図8


Tire_1
Body_1
Chassis_1
Car_A
Door_1
Engine_1
Tire_2
Body_2
Chassis_1
Car_B
Door_2
Engine_1

[図9]

図9

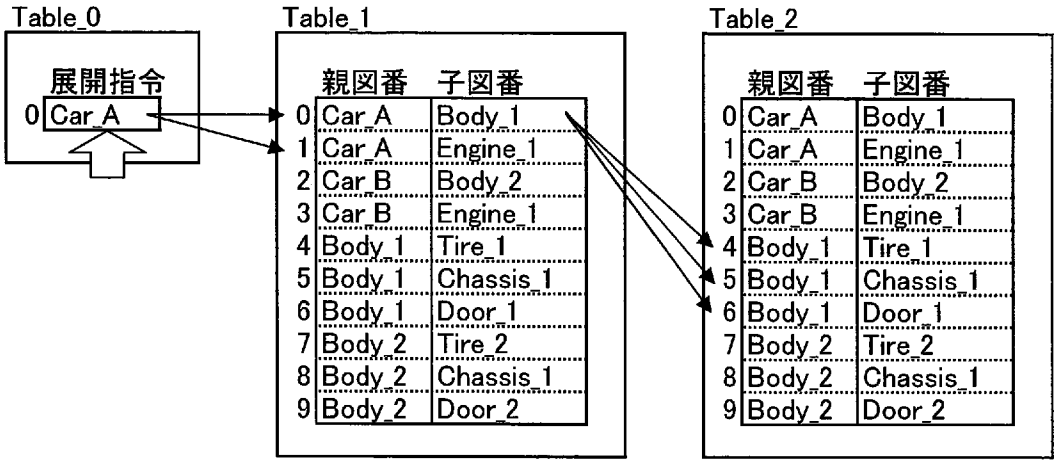


Table_0
展開指令
0 Car_A
Table_1
親図番 子図番
0 Car_A Body_1
1 Car_A Engine_1
2 Car_B Body_2
3 Car_B Engine_1
4 Body_1 Tire_1
5 Body_1 Chassis_1
6 Body_1 Door_1
7 Body_2 Tire_2
8 Body_2 Chassis_1
9 Body_2 Door_2
Table_2
親図番 子図番
0 Car_A Body_1
1 Car_A Engine_1
2 Car_B Body_2
3 Car_B Engine_1
4 Body_1 Tire_1
5 Body_1 Chassis_1
6 Body_1 Door_1
7 Body_2 Tire_2
8 Body_2 Chassis_1
9 Body_2 Door_2

[図10]

図10A

	Topology	Title	Value
0	-1	展開指令	Car_A
1	0	子図番	Body_1
2	1	子図番	Tire_1
3	1	子図番	Chassis_1
4	1	子図番	Door_1
5	0	子図番	Engine_1

図10B

	Topology	Title	Value
0	-1	展開指令	Car_A
1	0	子図番	Body_1
2	0	子図番	Engine_1
3	1	子図番	Tire_1
4	1	子図番	Chassis_1
5	1	子図番	Door_1

[図11]

図11A

	Topology
0	-1
1	0
2	1
3	1
4	1
5	0

Title

	VNo
0	1
1	0
2	0
3	0
4	0
5	0

	VL
0	子図番
1	展開指令

Value

	VNo
0	2
1	0
2	8
3	4
4	5
5	7

	VL
0	Body_1
1	Body_2
2	Car_A
3	Car_B
4	Chassis_1
5	Door_1
6	Door_2
7	Engine_1
8	Tire_1
9	Tire_2

図11B

	Topology
0	-1
1	0
2	0
3	1
4	1
5	1

Title

	VNo
0	1
1	0
2	0
3	0
4	0
5	0

	VL
0	子図番
1	展開指令

Value

	VNo
0	2
1	0
2	7
3	8
4	4
5	5

	VL
0	Body_1
1	Body_2
2	Car_A
3	Car_B
4	Chassis_1
5	Door_1
6	Door_2
7	Engine_1
8	Tire_1
9	Tire_2

[図12]

図12A

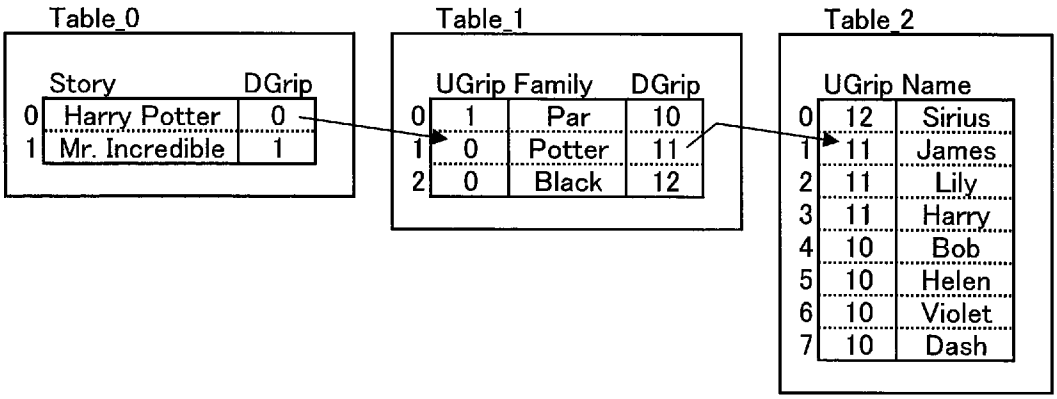
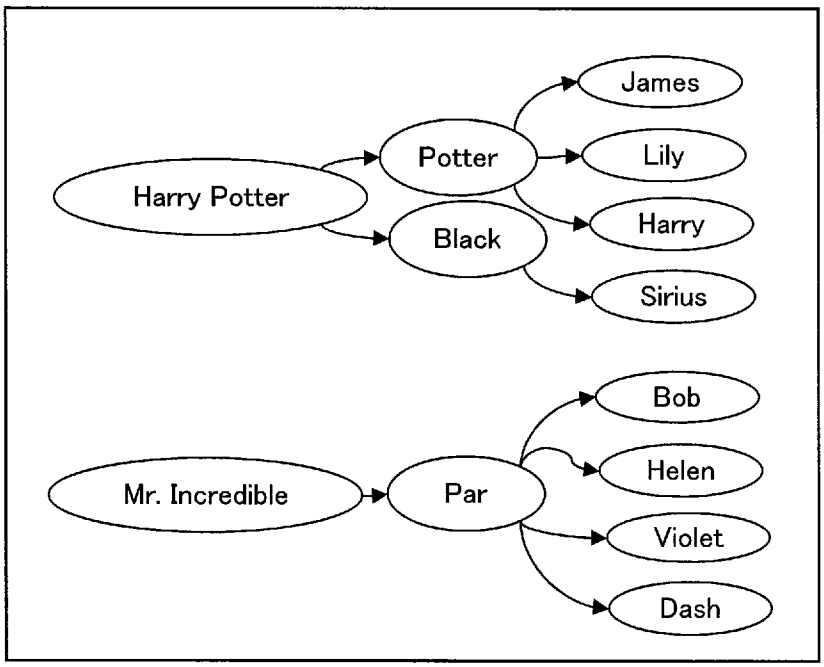


図12B



[図13]

図13A

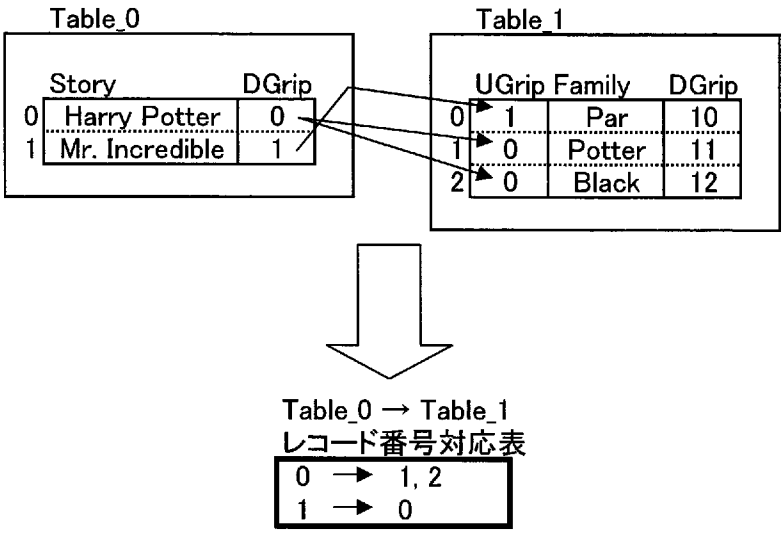
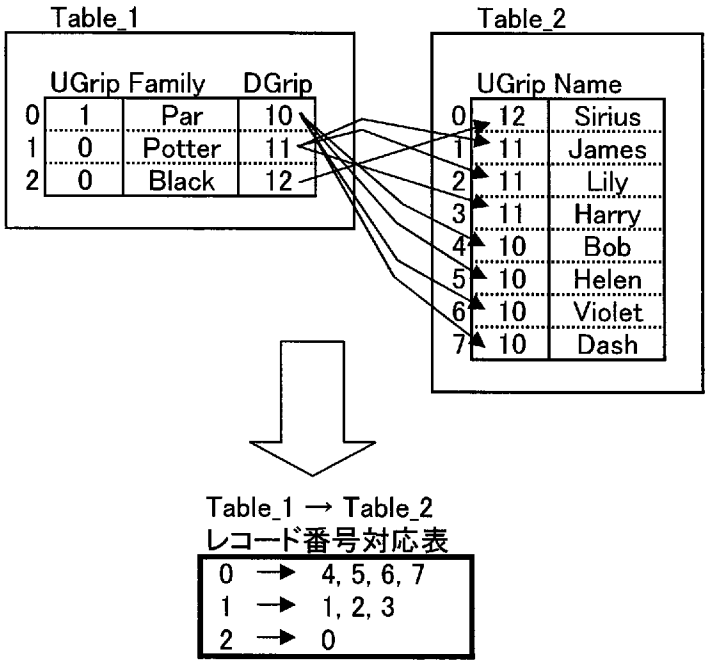


図13B



[図14]

図14A

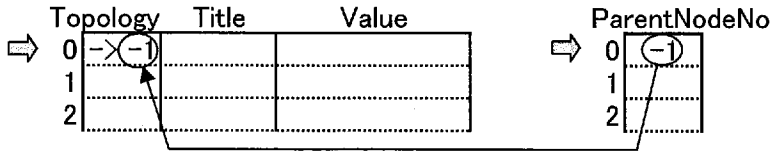


図14B

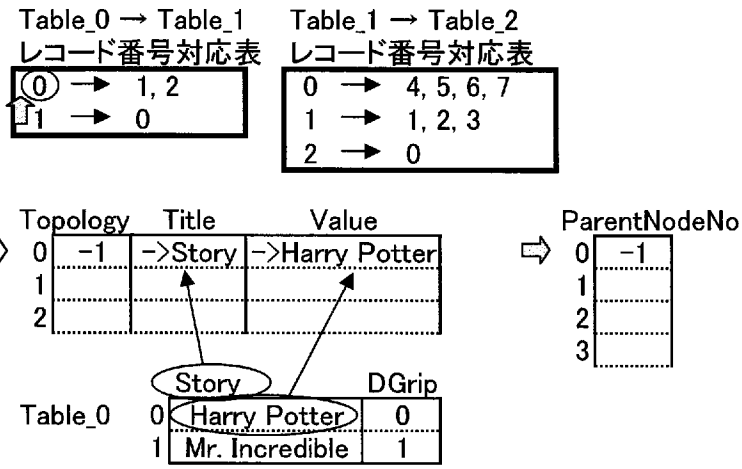
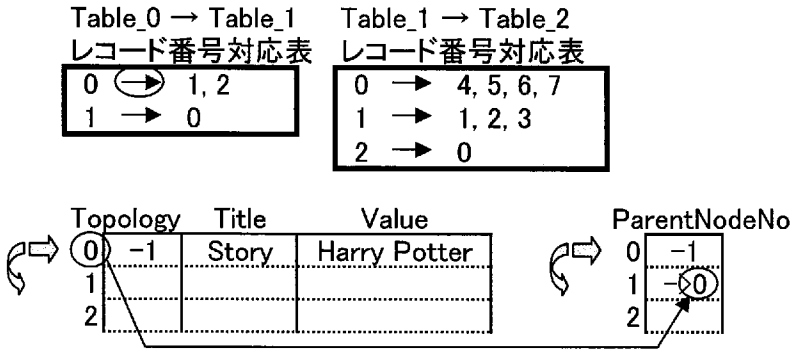


図14C



[図15]

図15A

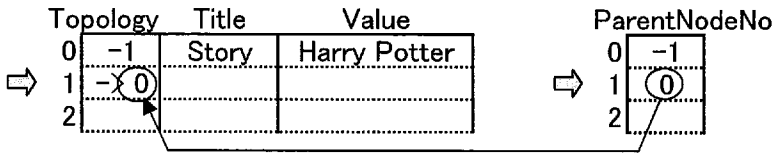


図15B

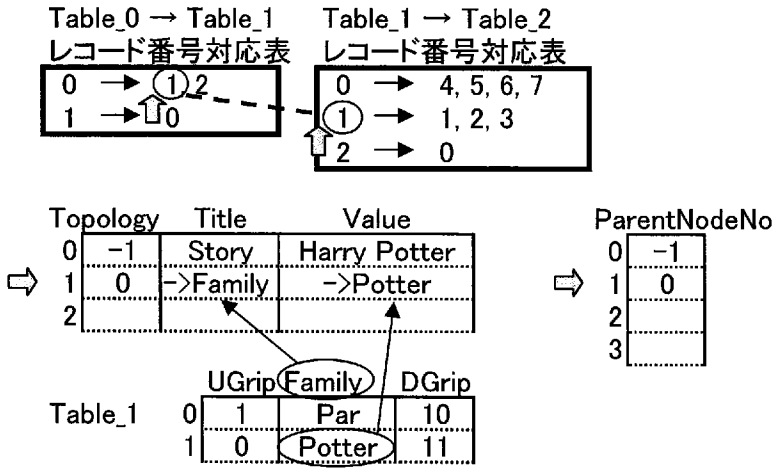
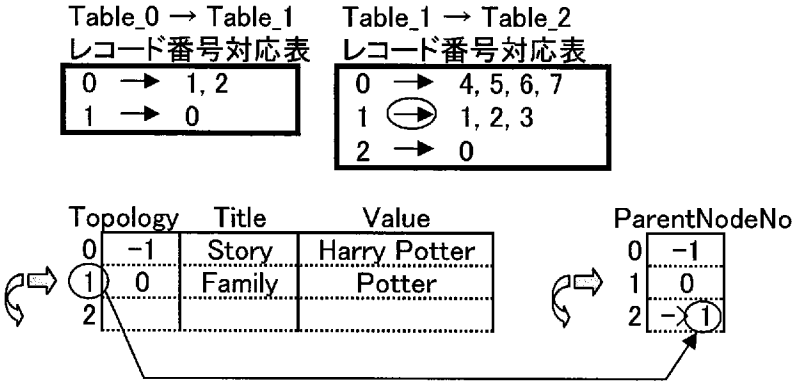


図15C



[図16]

図16A

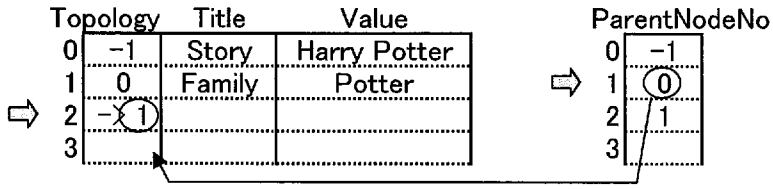


図16B

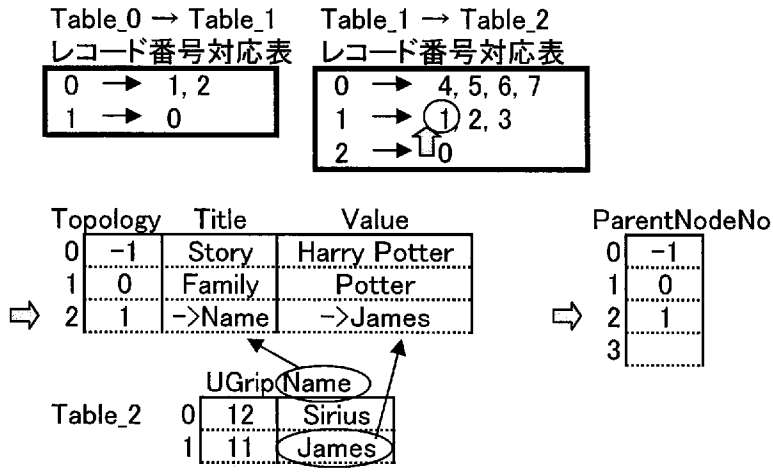
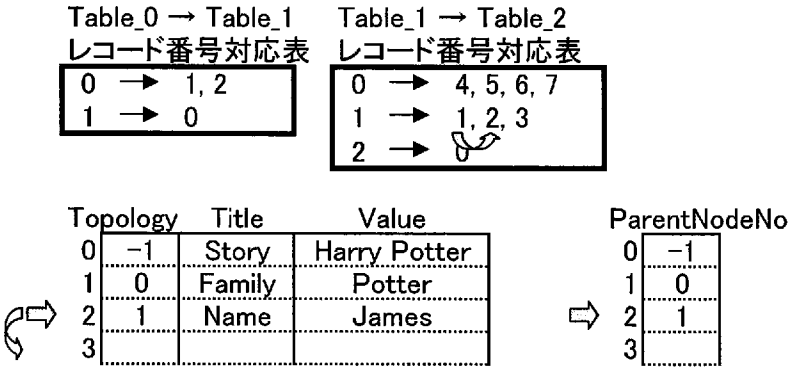


図16C



[図17]

図17A

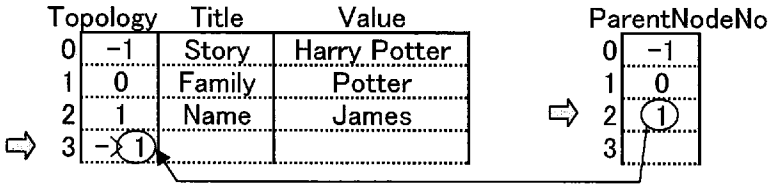


図17B

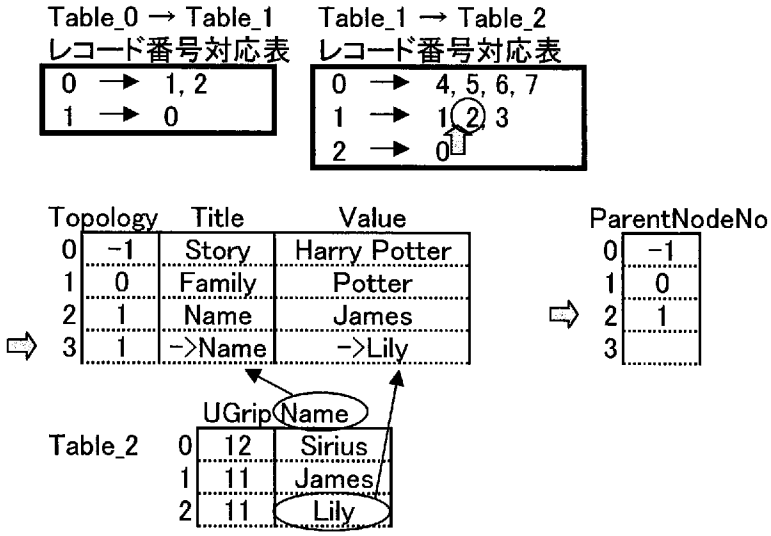
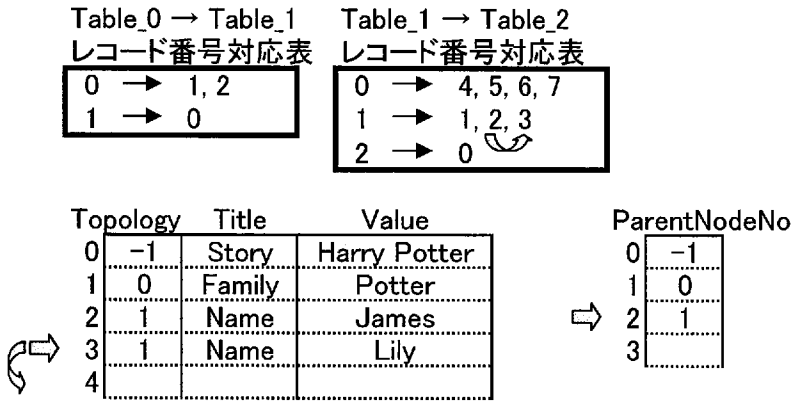


図17C



[図18]

図18A

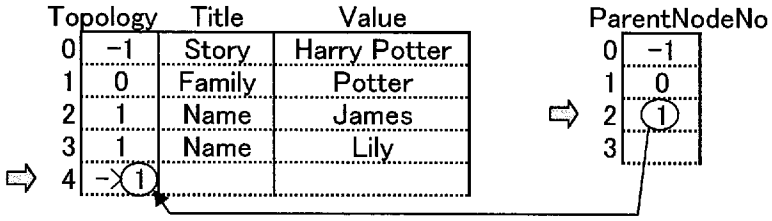


図18B

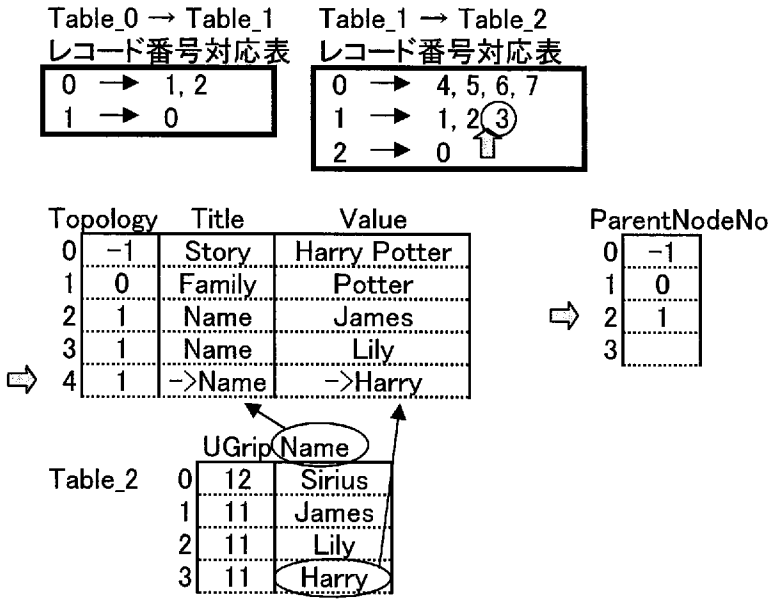
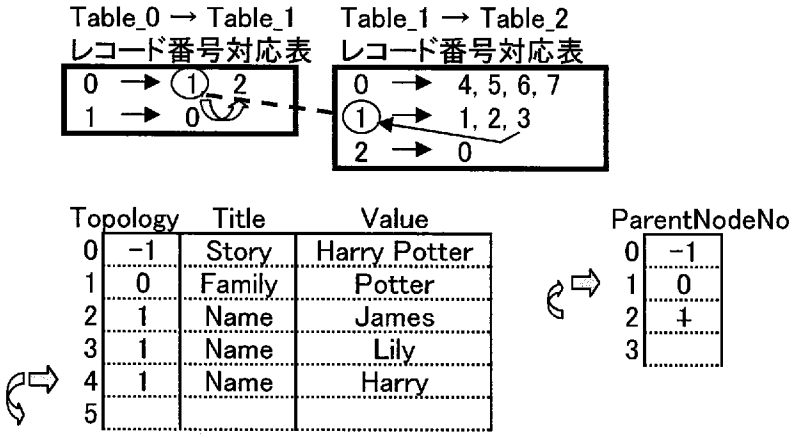


図18C



[図19]

図19A

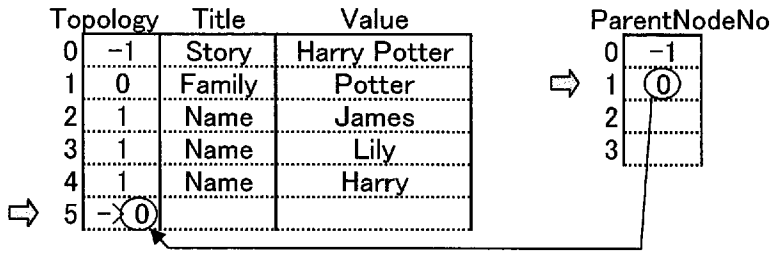


図19B

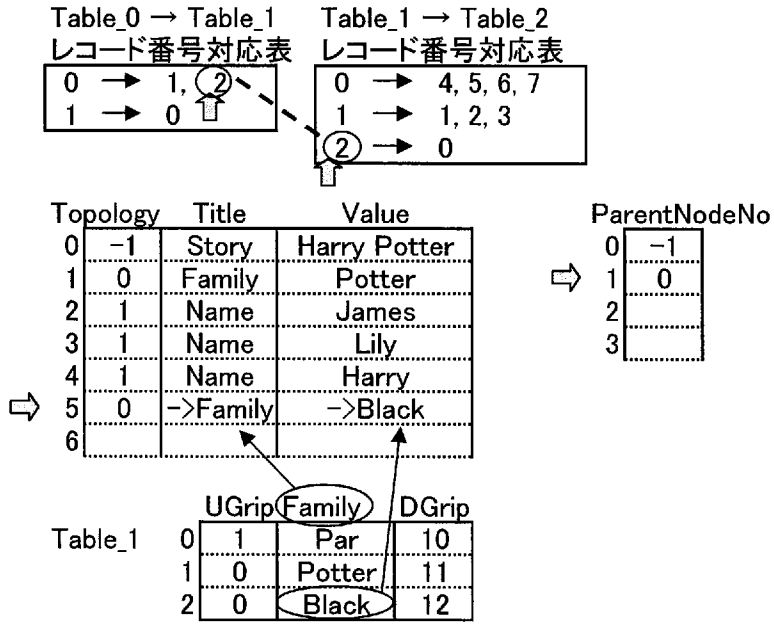
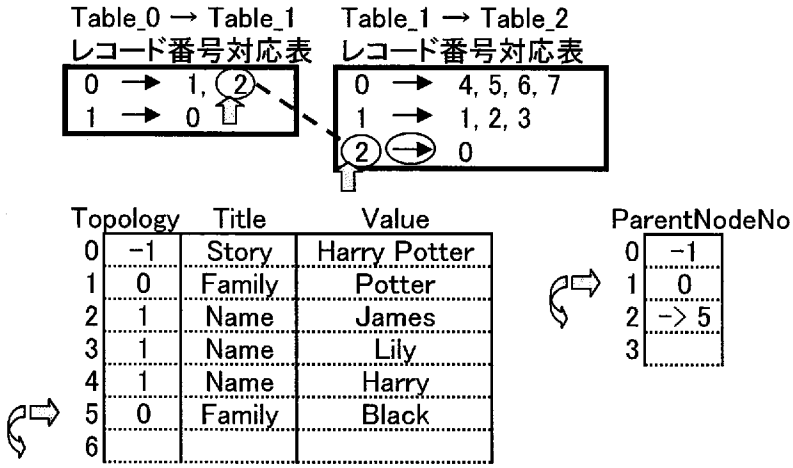


図19C



[図20]

図20A

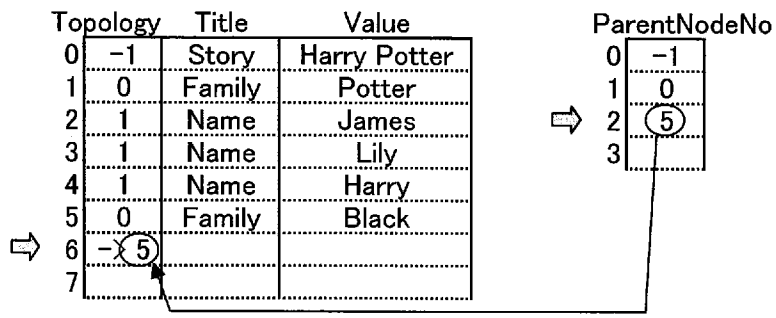


図20B

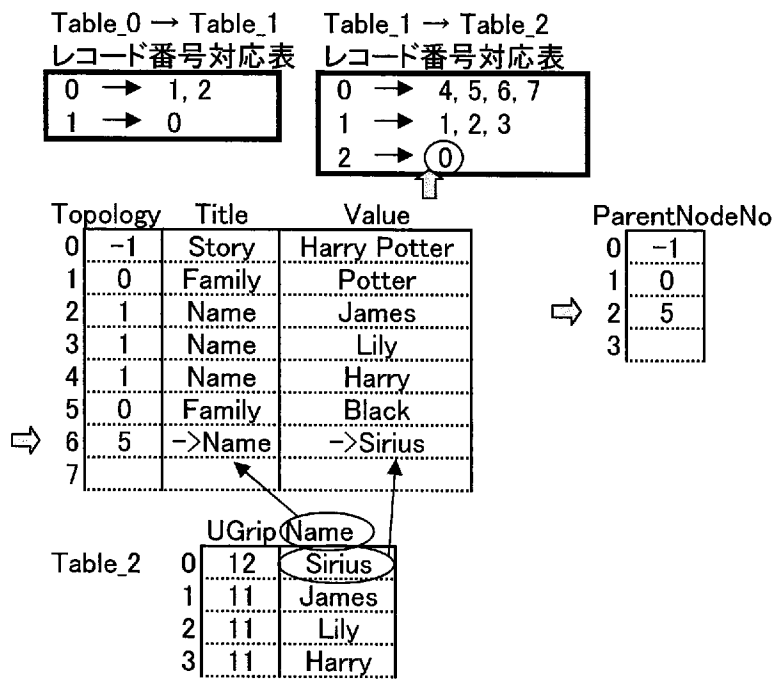
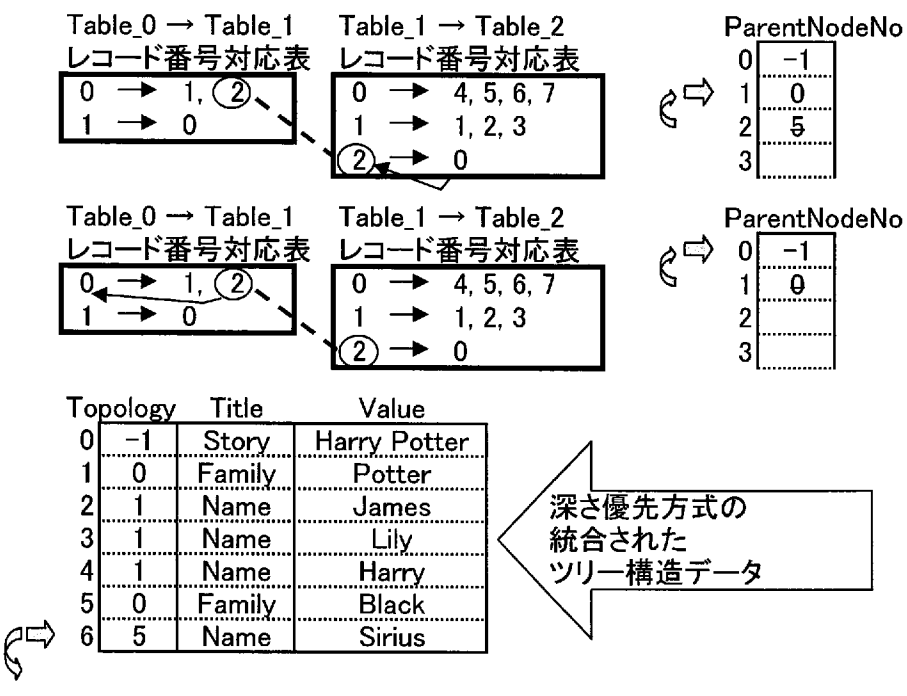


図20C



[図21]

図21A

	Topology	Title	Value
0	-1	Story	Harry Potter
1	0	Family	Potter
2	1	Name	James
3	1	Name	Lily
4	1	Name	Harry
5	0	Family	Black
6	5	Name	Sirius

図21B

	Topology	Title (VNo)	Value (VNo)
0	-1	2	4
1	0	0	10
2	1	1	6
3	1	1	7
4	1	1	3
5	0	0	0
6	5	1	11

(値番号) (値番号)

Title の VL

0	Family
1	Name
2	Story

(値リスト)

Value の VL

0	Black
1	Bob
2	Dash
3	Harry
4	Harry Potter
5	Helen
6	James
7	Lily
8	Mr. Incredible
9	Par
10	Potter
11	Sirius
12	Violet

(値リスト)

図21C

	Topology
0	-1
1	0
2	1
3	1
4	1
5	0
6	5

Title

	VNo
	2
	0
	1
	1
	1
	0
	1

(値番号)

	VL
0	Family
1	Name
2	Story

(値リスト)

Value

	VNo
	4
	10
	6
	7
	3
	0
	11

(値番号)

	VL
0	Black
1	Bob
2	Dash
3	Harry
4	Harry Potter
5	Helen
6	James
7	Lily
8	Mr. Incredible
9	Par
10	Potter
11	Sirius
12	Violet

(値リスト)

[図22]

図22

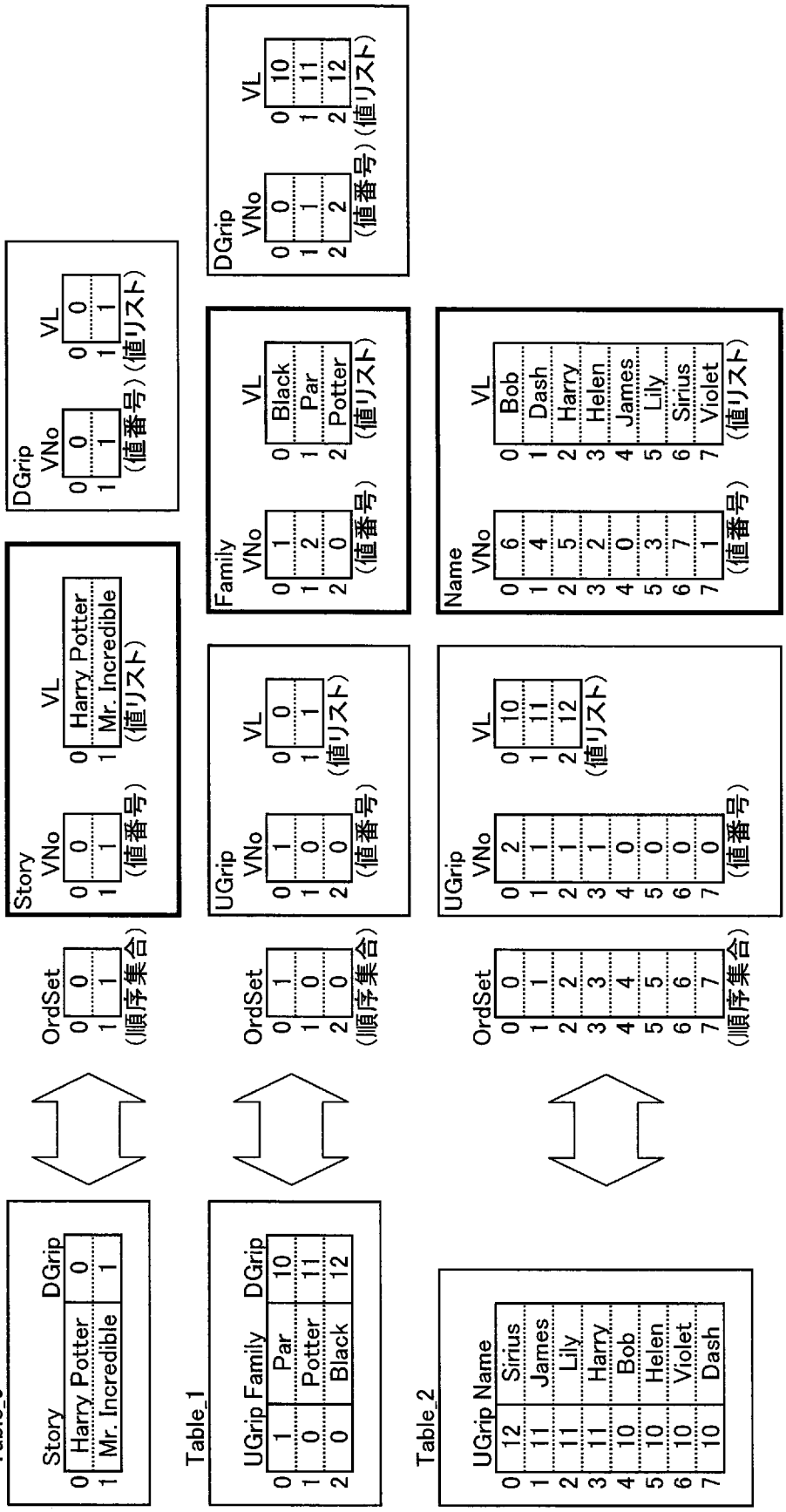


Table_0
Story
DGrip
Harry Potter
Mr. Incredible
Table_1
UGrip
Family
DGrip
Par
Potter
Black
Table_2
UGrip
Name
Sirius
James
Lily
Harry
Bob
Helen
Violet
Dash
OrdSet
(順序集合)
Story
VNo
VL
(値番号)
(値リスト)
DGrip
UGrip
Family
Name
Harry Potter
Mr. Incredible
Black
Bob
Dash
James
Lily

[図23]

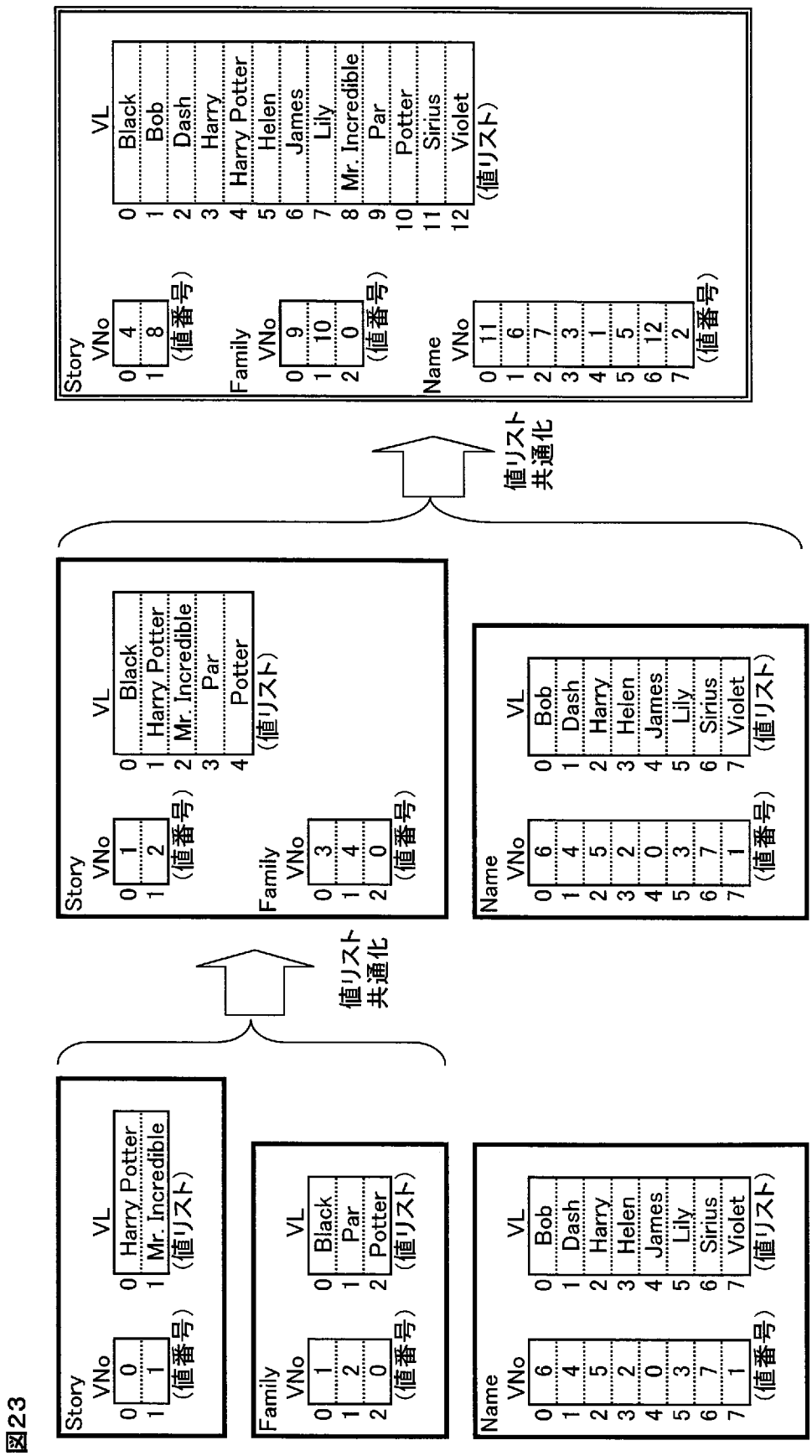
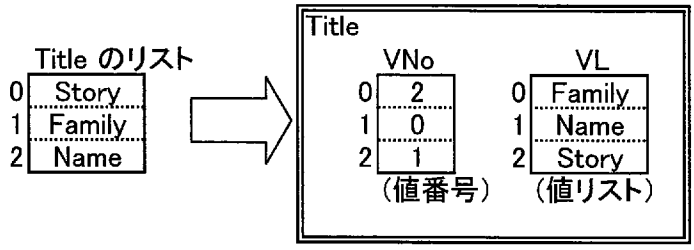

Story
VNo
VL
(値番号)
(値リスト)
Family
Name
値リスト
共通化
Harry Potter
Mr. Incredible
Black
Par
Potter
Bob
Dash
Harry
Helen
James
Lily
Sirius
Violet

図23

[図24]

図24


Title のリスト
0 Story
1 Family
2 Name
Title
VNo
0 2
1 0
2 1
（値番号）
VL
0 Family
1 Name
2 Story
（値リスト）

[図25]

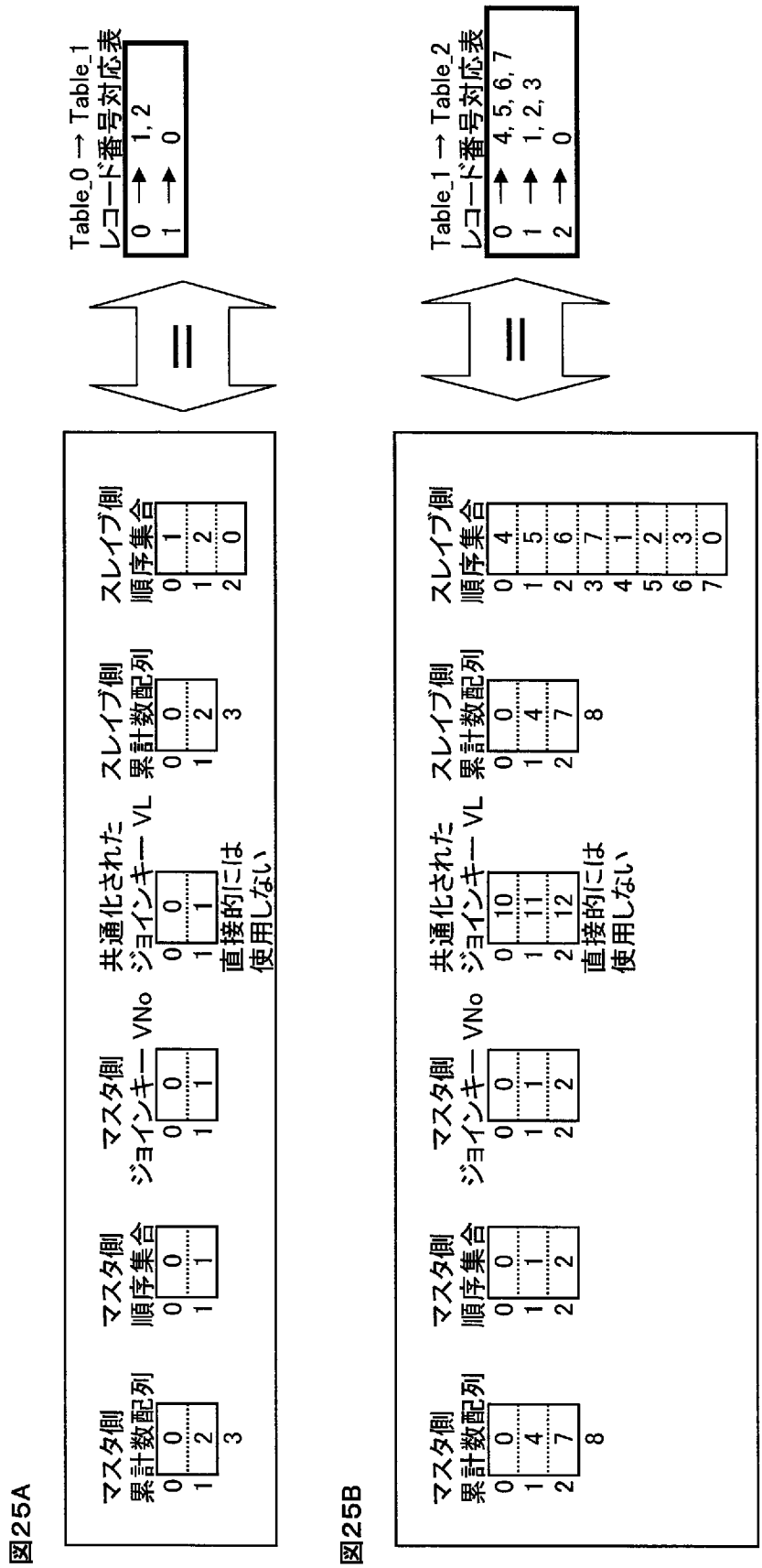


図25A
マスタ側
累計数配列
0
1
2
3
マスタ側
順序集合
0
1
マスタ側
ジョインキー VNo
0
1
共通化された
ジョインキー VL
0
1
直接的には
使用しない
スレイブ側
累計数配列
0
1
2
3
スレイブ側
順序集合
0
1
2
Table_0 → Table_1
レコード番号対応表
0 → 1, 2
1 → 0
図25B
マスタ側
累計数配列
0
1
2
8
マスタ側
順序集合
0
1
2
マスタ側
ジョインキー VNo
0
1
2
共通化された
ジョインキー VL
0
1
2
直接的には
使用しない
スレイブ側
累計数配列
0
1
2
8
スレイブ側
順序集合
0
1
2
3
4
5
6
7
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 → 1, 2, 3
2 → 0

[図26]

図26A

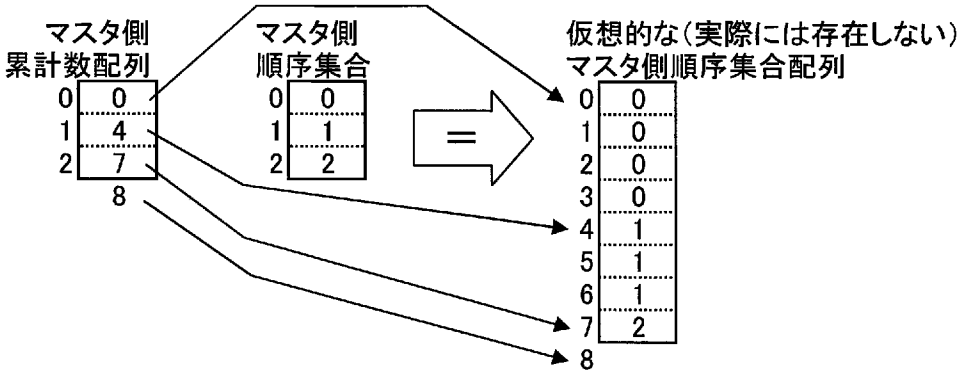
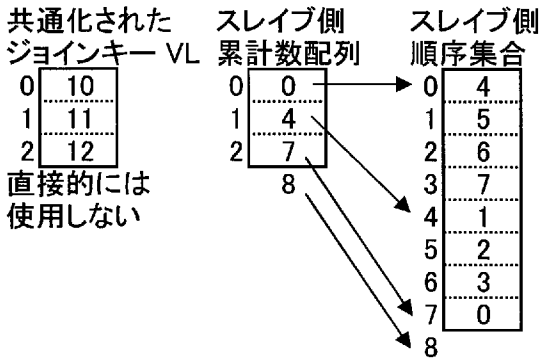


図26B



[図27]

図27

Tree_0

	Topology	Title	Value
0	-1	Story	Harry Potter
1	0	Family	Potter
2	1	Name	James
3	1	Name	Lily
4	1	Name	Harry
5	0	Family	Black
6	5	Name	Sirius

Tree_1

	Topology	Title	Value
0	-1	Story	Mr. Incredible
1	0	Family	Par
2	1	Name	Bob
3	1	Name	Helen
4	1	Name	Violet
6	1	Name	Dash

[図28]

図28

	Topology	Title	Value
0	-1	-	-
1	0	Story	Harry Potter
2	1	Family	Potter
3	2	Name	James
4	2	Name	Lily
5	2	Name	Harry
6	1	Family	Black
7	6	Name	Sirius
8	0	Story	Mr. Incredible
9	8	Family	Par
10	9	Name	Bob
11	9	Name	Helen
12	9	Name	Violet
13	9	Name	Dash

[図29]

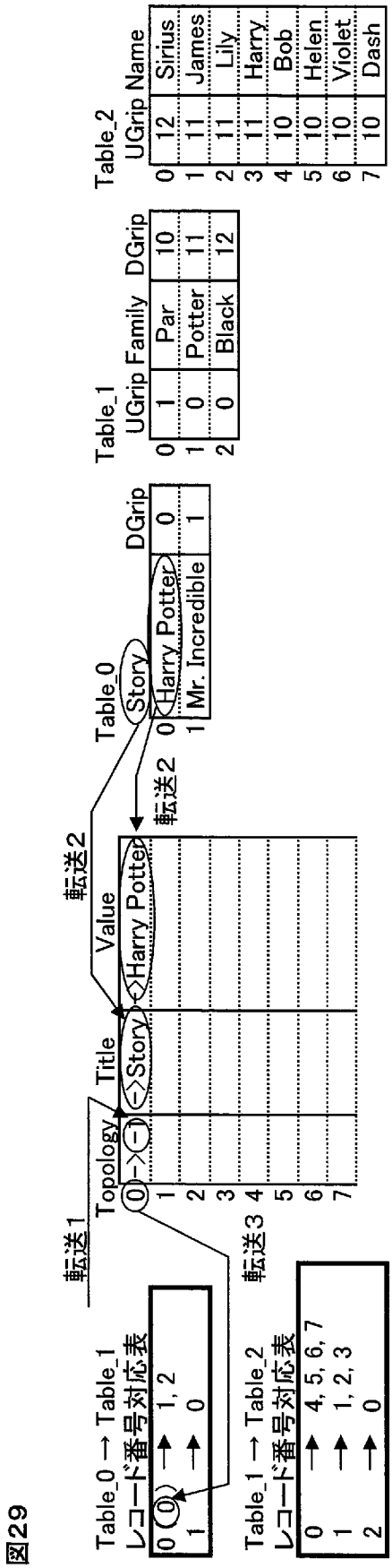


図29
転送1
Table_0 → Table_1
レコード番号対応表
0 → 1, 2
1 → 0
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 → 1, 2, 3
2 → 0
転送3
Topology
Title
Value
0->-1
->Story
Harry Potter
転送2
Table_0
Story
DGrip
Harry Potter 0
Mr. Incredible 1
転送2
Table_1
UGrip Family DGrip
1 Par 10
0 Potter 11
0 Black 12
Table_2
UGrip Name
12 Sirius
11 James
11 Lily
11 Harry
10 Bob
10 Helen
10 Violet
10 Dash

[図30]

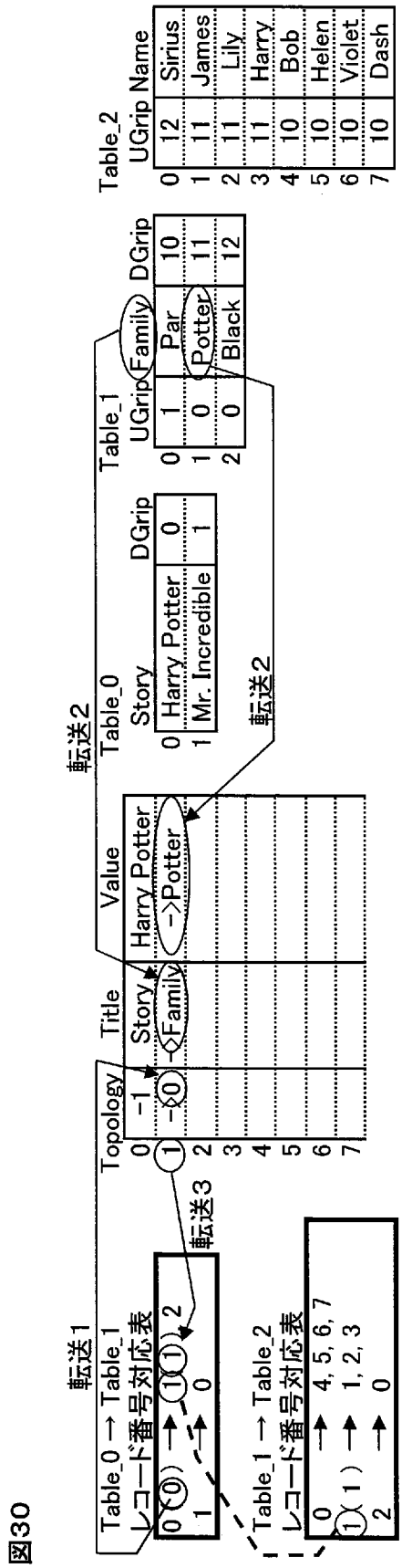


図30
転送1
Table_0 → Table_1
レコード番号対応表
0 (0) → (1) 2
1 → 0
転送3
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
(1) → 1, 2, 3
2 → 0
転送2
Topology
Title
Value
-1
Story
Harry Potter
0
Family
->Potter
1
転送2
Table_0
Story
DGrip
0 Harry Potter 0
1 Mr. Incredible 1
Table_1
UGrip
Family
DGrip
0 1 Par 10
1 0 Potter 11
2 0 Black 12
Table_2
UGrip
Name
0 12 Sirius
1 11 James
2 11 Lily
3 11 Harry
4 10 Bob
5 10 Helen
6 10 Violet
7 10 Dash

[図31]

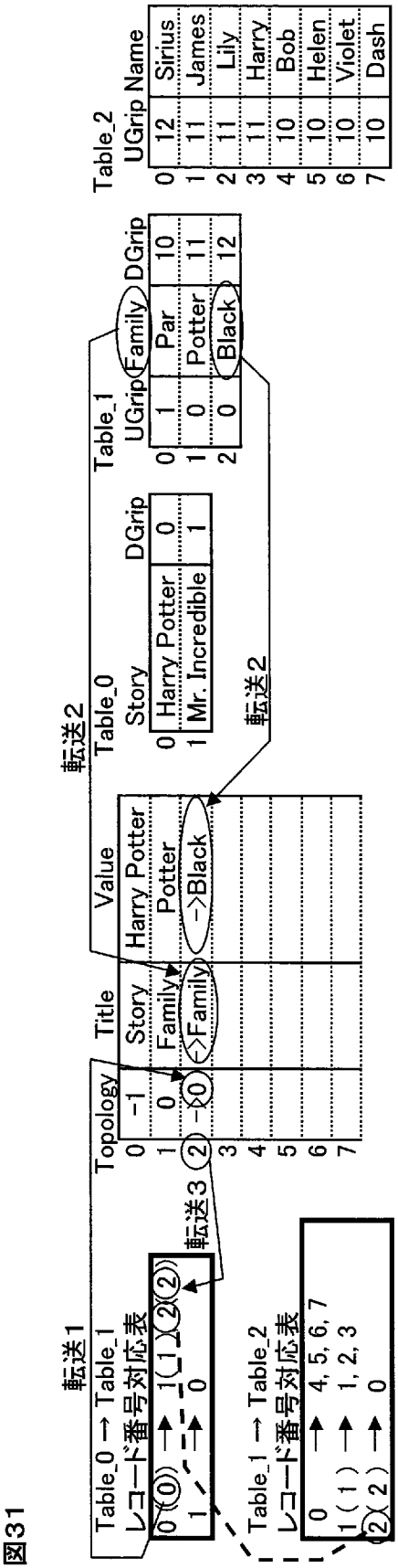


図31
転送1
Table_0 → Table_1
レコード番号対応表
0 (0) → 1 (1) (2)
1 → 0
転送3
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 (1) → 1, 2, 3
(2) 2) → 0
転送2
Table_0
Topology
Title
Value
-1
Story
Harry Potter
0
Family
Potter
2
->0
<Family
->Black
Story
DGrip
0
Harry Potter
0
1
Mr. Incredible
1
転送2
Table_1
UGrip
Family
DGrip
0
1
Par
10
1
0
Potter
11
2
0
Black
12
Table_2
UGrip
Name
0
12
Sirius
1
11
James
2
11
Lily
3
11
Harry
4
10
Bob
5
10
Helen
6
10
Violet
7
10
Dash

[図32]

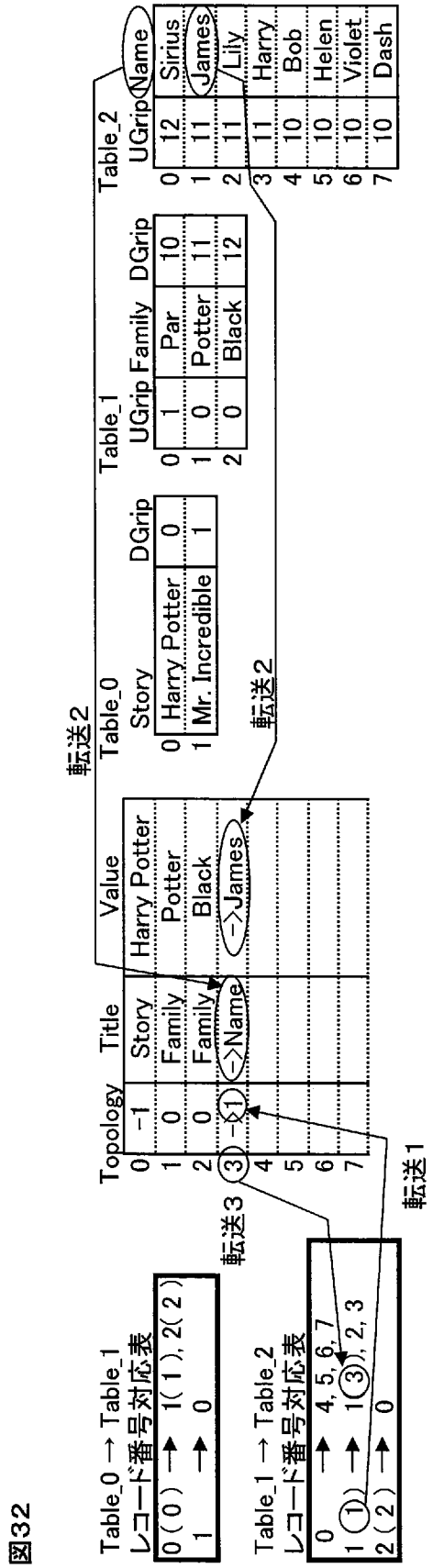


図32
Table_0 → Table_1
レコード番号対応表
0 (0) → 1(1), 2(2)
1 → 0
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 (1) → 1(3), 2, 3
2 (2) → 0
転送1
転送2
転送3
転送2
Topology
Title
Value
Table_0
Story
DGrip
Harry Potter
Mr. Incredible
Table_1
UGrip
Family
DGrip
Par
Potter
Black
Table_2
UGrip
Name
Sirius
James
Lily
Harry
Bob
Helen
Violet
Dash
Story
Family
Family
->Name
->James

[図33]

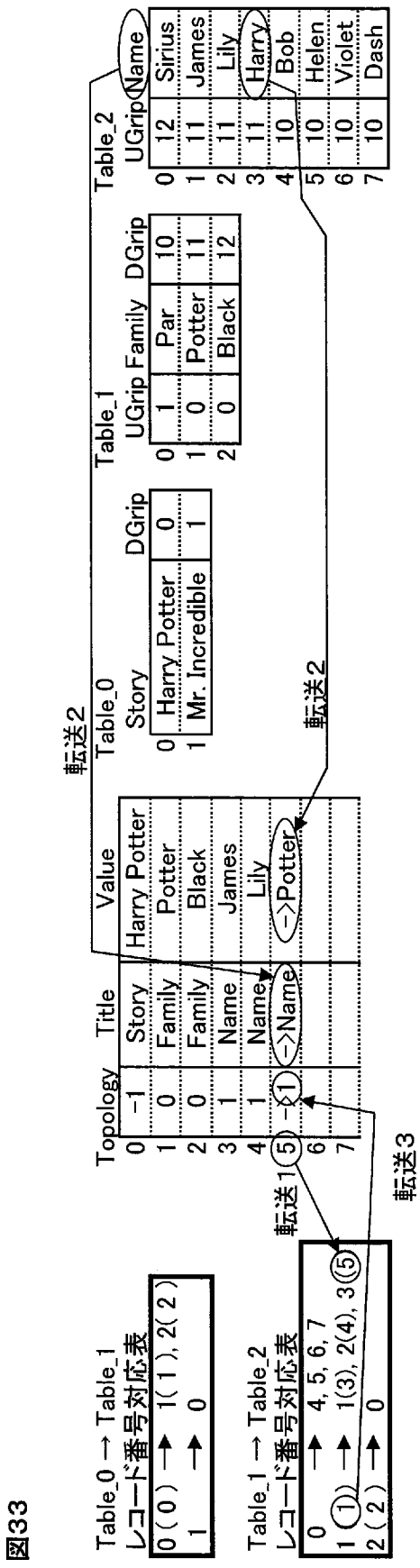


図33
Table_0 → Table_1
レコード番号対応表
0 (0) → 1(1), 2(2)
1 → 0
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 (1) → 1(3), 2(4), 3(5)
2 (2) → 0
転送1
転送2
転送3
Topology
Title
Value
Story
Family
Family
Name
Name
->Name
Harry Potter
Potter
Black
James
Lily
->Potter
Table_0
Story
DGrip
Harry Potter
0
Mr. Incredible
1
Table_1
UGrip
Family
DGrip
1
Par
10
0
Potter
11
2
Black
12
Table_2
UGrip
Name
12
Sirius
11
James
11
Lily
11
Harry
10
Bob
10
Helen
10
Violet
10
Dash

[図34]

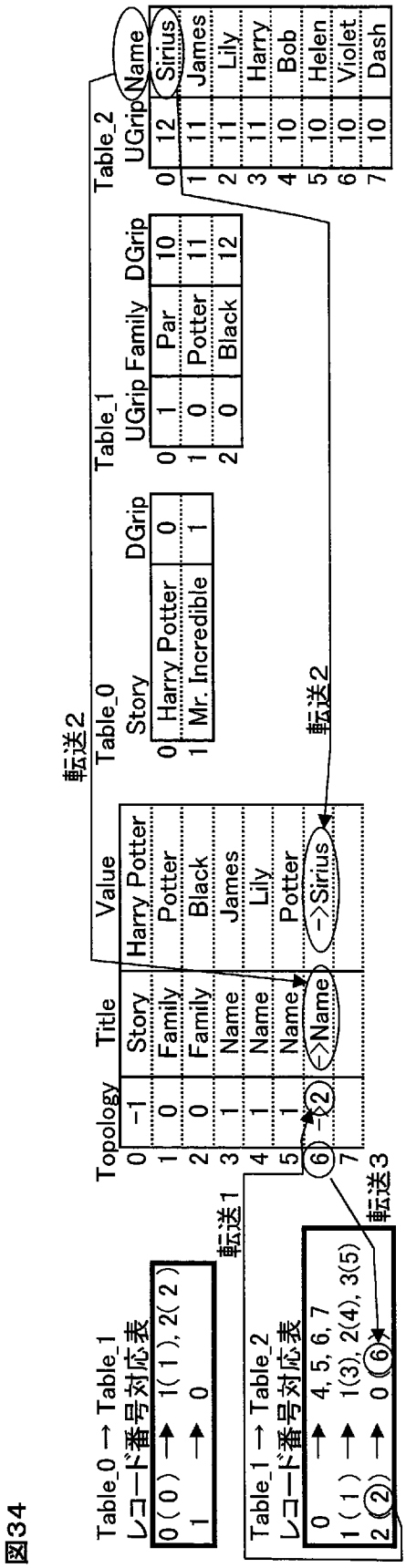
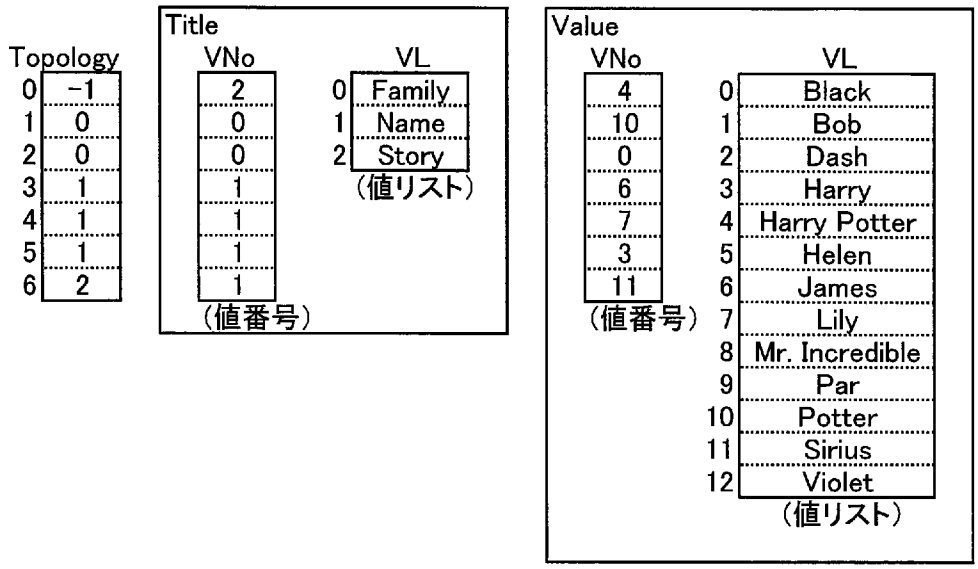


図34
Table_0 → Table_1
レコード番号対応表
0 (0) → 1(1), 2(2)
1 → 0
Table_1 → Table_2
レコード番号対応表
0 → 4, 5, 6, 7
1 (1) → 1(3), 2(4), 3(5)
2 (2) → 0 (6)
転送1
転送2
転送3
Topology
Title
Value
Table_0
Story
DGrip
Harry Potter
Mr. Incredible
Table_1
UGrip
Family
Par
Potter
Black
Table_2
UGrip
Name
Sirius
James
Lily
Harry
Bob
Helen
Violet
Dash
Story
Family
Family
Name
Name
Name
Harry Potter
Potter
Black
James
Lily
Potter
->Name
->Sirius

[図35]

図35



[図36]

図36

Tree_0

	Topology	Title	Value
0	-1	Story	Harry Potter
1	0	Family	Potter
2	0	Family	Black
3	1	Name	James
4	1	Name	Lily
5	1	Name	Harry
6	2	Name	Sirius

Tree_1

	Topology	Title	Value
0	-1	Story	Mr. Incredible
1	0	Family	Par
2	1	Name	Bob
3	1	Name	Helen
4	1	Name	Violet
6	1	Name	Dash

[図37]

図37

	Topology	Title	Value
0	-1	-	-
1	0	Story	Harry Potter
2	0	Story	Mr. Incredible
3	1	Family	Potter
4	1	Family	Black
5	2	Family	Par
6	3	Name	James
7	3	Name	Lily
8	3	Name	Harry
9	4	Name	Sirius
10	5	Name	Bob
11	5	Name	Helen
12	5	Name	Violet
13	5	Name	Dash

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2008/051185

A. CLASSIFICATION OF SUBJECT MATTER
G06F17/30(2006.01)i, *G06F12/00*(2006.01)i, *G06F17/21*(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
G06F17/30, G06F12/00, G06F17/21

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1922-1996 Jitsuyo Shinan Toroku Koho 1996-2008
Kokai Jitsuyo Shinan Koho 1971-2008 Toroku Jitsuyo Shinan Koho 1994-2008

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
NRI Cyber Patent

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2000-267906 A (Mitsubishi Electric Corp.), 29 September, 2000 (29.09.00), Full text; Figs. 1 to 4 (Family: none)	1-18
A	JP 2003-67403 A (Fuji Xerox Co., Ltd.), 07 March, 2003 (07.03.03), Full text; Figs. 1 to 17 & US 2003/0159110 A1	1-18
A	JP 2004-310249 A (Systems Engineering Consultants Co., Ltd.), 04 November, 2004 (04.11.04), Full text; Figs. 1 to 8 (Family: none)	1-18

☒ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
25 February, 2008 (25.02.08)	11 March, 2008 (11.03.08)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2008/051185

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
A	JP 2006-114045 A (Microsoft Corp.), 27 April, 2006 (27.04.06), Full text; Figs. 1 to 12 & US 2006/0085451 A1 & EP 1647905 A1	1-18
A	Yoshifumi MASUNAGA, Relational Database Nyumon -Data Model·SQL·Kanri System-, first edition, Saiensu-sha Co., Ltd., 25 January, 1991 (25.01.91), pages 145 to 148	1-18

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP2008/051185

Box No. II Observations where certain claims were found unsearchable (Continuation of item 2 of first sheet)

This international search report has not been established in respect of certain claims under Article 17(2)(a) for the following reasons:

1. [X] Claims Nos.: 19, 20
because they relate to subject matter not required to be searched by this Authority, namely:
Claims 19, 20 relate to tree-structured data recorded on a computer-readable medium, which relates to mere presentations of information, i.e., an object not requiring search by the International Search Authority under the provisions stipulated in PCT Article 17 (2)(a)(i) and PCT Rule 39.1(v).

2. [ ] Claims Nos.:
because they relate to parts of the international application that do not comply with the prescribed requirements to such an extent that no meaningful international search can be carried out, specifically:

3. [ ] Claims Nos.:
because they are dependent claims and are not drafted in accordance with the second and third sentences of Rule 6.4(a).

Box No. III Observations where unity of invention is lacking (Continuation of item 3 of first sheet)

This International Searching Authority found multiple inventions in this international application, as follows:

1. [ ] As all required additional search fees were timely paid by the applicant, this international search report covers all searchable claims.

2. [ ] As all searchable claims could be searched without effort justifying additional fees, this Authority did not invite payment of additional fees.

3. [ ] As only some of the required additional search fees were timely paid by the applicant, this international search report covers only those claims for which fees were paid, specifically claims Nos.:

4. [ ] No required additional search fees were timely paid by the applicant. Consequently, this international search report is restricted to the invention first mentioned in the claims; it is covered by claims Nos.:

Remark on Protest

the

[ ] The additional search fees were accompanied by the applicant's protest and, where applicable, payment of a protest fee.

[ ] The additional search fees were accompanied by the applicant's protest but the applicable protest fee was not paid within the time limit specified in the invitation.

[ ] No protest accompanied the payment of additional search fees.

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００８／０５１１８５

Ａ．　発明の属する分野の分類（国際特許分類（ＩＰＣ））

Int.Cl.　G06F17/30(2006.01)i, G06F12/00(2006.01)i, G06F17/21(2006.01)i

Ｂ．　調査を行った分野

調査を行った最小限資料（国際特許分類（ＩＰＣ））

Int.Cl.　G06F17/30, G06F12/00, G06F17/21

最小限資料以外の資料で調査を行った分野に含まれるもの

日本国実用新案公報	１９２２－１９９６年
日本国公開実用新案公報	１９７１－２００８年
日本国実用新案登録公報	１９９６－２００８年
日本国登録実用新案公報	１９９４－２００８年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

NRI サイバーパテント

Ｃ．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	JP 2000-267906 A（三菱電機株式会社）2000.09.29, 全文, 第１－４図（ファミリーなし）	1-18
A	JP 2003-67403 A（富士ゼロックス株式会社）2003.03.07, 全文, 第１－１７図 & US 2003/0159110 A1	1-18
A	JP 2004-310249 A（株式会社セック）2004.11.04, 全文, 第１－８図（ファミリーなし）	1-18

☑　Ｃ欄の続きにも文献が列挙されている。　　☐　パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「Ａ」特に関連のある文献ではなく、一般的技術水準を示すもの
「Ｅ」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「Ｌ」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「Ｏ」口頭による開示、使用、展示等に言及する文献
「Ｐ」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「Ｔ」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「Ｘ」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Ｙ」特に関連のある文献であって、当該文献と他の１以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「＆」同一パテントファミリー文献

国際調査を完了した日 ２５．０２．２００８	国際調査報告の発送日 １１．０３．２００８
国際調査機関の名称及びあて先 日本国特許庁（ＩＳＡ／ＪＰ） 郵便番号１００－８９１５ 東京都千代田区霞が関三丁目４番３号	特許庁審査官（権限のある職員）　５Ｍ　３２５２ 鈴木　和樹 電話番号　０３－３５８１－１１０１　内線　３５９９

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
A	JP 2006-114045 A (マイクロソフト　コーポレーション) 2006.04.27, 全文, 第１－１２図 & US 2006/0085451 A1 & EP 1647905 A1	1-18
A	増永良文, リレーショナルデータベース入門　－データモデル・ＳＱＬ・管理システム－, 初版, 株式会社サイエンス社, 1991.01.25, ｐ．１４５－１４８	1-18

国際調査報告

国際出願番号　ＰＣＴ／ＪＰ２００８／０５１１８５

第Ⅱ欄　請求の範囲の一部の調査ができないときの意見（第１ページの２の続き）

法第8条第3項（ＰＣＴ17条(2)(a)）の規定により、この国際調査報告は次の理由により請求の範囲の一部について作成しなかった。

１．☑ 請求の範囲１９，２０ は、この国際調査機関が調査をすることを要しない対象に係るものである。
つまり、
請求の範囲１９，２０は、コンピュータ読み取り可能な記録媒体にツリー構造データが記録されているだけであり、情報の単なる提示に該当し、ＰＣＴ第１７条（２）（ａ）（ｉ）及びＰＣＴ規則３９．１（ｖ）の規定により、この国際調査機関が調査することを要しない対象に係るものである。

２．☐ 請求の範囲＿＿＿＿＿＿＿＿ は、有意義な国際調査をすることができる程度まで所定の要件を満たしていない国際出願の部分に係るものである。つまり、

３．☐ 請求の範囲＿＿＿＿＿＿＿＿ は、従属請求の範囲であってＰＣＴ規則6.4(a)の第２文及び第3文の規定に従って記載されていない。

第Ⅲ欄　発明の単一性が欠如しているときの意見（第１ページの３の続き）

次に述べるようにこの国際出願に二以上の発明があるとこの国際調査機関は認めた。

１．☐ 出願人が必要な追加調査手数料をすべて期間内に納付したので、この国際調査報告は、すべての調査可能な請求の範囲について作成した。

２．☐ 追加調査手数料を要求するまでもなく、すべての調査可能な請求の範囲について調査することができたので、追加調査手数料の納付を求めなかった。

３．☐ 出願人が必要な追加調査手数料を一部のみしか期間内に納付しなかったので、この国際調査報告は、手数料の納付のあった次の請求の範囲のみについて作成した。

４．☐ 出願人が必要な追加調査手数料を期間内に納付しなかったので、この国際調査報告は、請求の範囲の最初に記載されている発明に係る次の請求の範囲について作成した。

追加調査手数料の異議の申立てに関する注意

☐ 追加調査手数料及び、該当する場合には、異議申立手数料の納付と共に、出願人から異議申立てがあった。

☐ 追加調査手数料の納付と共に出願人から異議申立てがあったが、異議申立手数料が納付命令書に示した期間内に支払われなかった。

☐ 追加調査手数料の納付はあったが、異議申立てはなかった。