数理解析研究所講究録
593巻 1986年 29-40

意味表現に適した非正規インタフェースの開発

九州大学·工学部　山本秀樹　(Hideki Yamamoto)
九州大学·工学部　上林弥彦　(Yahiko Kambayashi)

1.まえがき

データベースシステムの応用分野の拡大にともない、利用者にわかりやすい出力がデータベースにも求められるようになった。現在広く用いられている関係データベースシステムでは、出力は第一正規形で表される。しかし、質問結果が満足する種々の意味制約のため、第一正規形による表現は冗長度の高いものになる可能性があり、利用者インタフェースとして適当ではないことが多い。

冗長度が低く直観的にわかりやすい、利用者インタフェースに適した出力方式として、出力に非正規形を用いる方法がある。これを非正規出力と呼ぶ。

非正規形は利用者インタフェースとして優れた面をもっているが、実際にデータベースシステムからの出力として用いるためには解決すべき問題点がある。その一つに、ある関係を第一正規形で表現する場合には表現の仕方は一通りしかないが、非正規形は同じ関係を複数の出力形式で表現できることがある。出力形式の選びかたによって出力のわかりやすさが大幅に異なることがあるので、どのような形式を用いるのが最もよいか、という問題は非常に重要である。非正規出力を実現する試みは以前から行われていたが[SLT]、出力形式の問題などの基本的な問題は解決していない。本稿では、非正規出力に関する問題を次の二つに分類して考察する。

(1) 冗長度が低く、利用者にわかりやすい出力をどのようにして得るか。
(2) 非正規出力の形式が与えられた時、非正規形を効率よく作るために、出力するための計算量をいかにして減らすか。

(1)の問題に対しては、利用者の質問とデータベース内の基底関係から最適な出力形式を作り、それを元に形式を決める方法を用いる。(2)の問題に対しては、質問を分解することによりデータベースシステムからの出力の冗長度を下げ、無駄な非正規化を減らすことにより効率の向上をはかる。最後に、これらの方法により既存の関係データベースシステムを用いて非正規出力を行う、非正規出力インタフェースの概要について述べる。

2.基本的事項

関係とは、一つ以上の定義域の直積の部分集合であり、直観的には表とみなすことができる。表の各行を組、各列を属性と呼ぶ。一つの関係Rを構成する属性の集合が$\{A_1, A_2, ..., A_n\}$であれば、その関係を$R(A_1, A_2, ..., A_n)$と書く。

各属性の定義域が単純値(集合ではない値)だけからなる関係を、正規化した関係(正規形)と言い、データベースに実際に蓄えられている関係を、基底関係と呼ぶ。データベースシステムに対する質問としては、選択、射影、結合からなる、関係代数質問(SPJ質問)を考える。意味制約として、関数従属性(FD)、結合従属性(JD)を考え、それぞれ次のような一般的な記法で表す。

FD : X→Y

JD : * [Y_1, Y_2, ..., Y_n]

関数従属性X→Yは属性Xの値を決めると属性Yの値が一意に決まることを表す。結合従属性は、属性集合$Y_1 \cup Y_2 \cup \cdots \cup Y_n$上の関係Rを$Y_i$, i=1, 2, ..., nのそれぞれの上へ射影を行い、その結果を再び結合したものがRに等しくなる必要十分条件である。

各属性の定義域が、集合値またはある属性集合上の関係、である関係を非正規化した関係(非正規形)と言う。非正規形を生成するための操作として、Group-byとRow-nestがある。

Group-by[X]は、属性集合Xの値の同じ部分に注目して関係の横切りを行い、その後Xの部分について同じ部分関係内の組をまとめる操作である。関係の横切りとは、一つの関係に含まれる組を分割し、それぞれの組集合をまとめて表示する操作である。Row-nest[X]は、X以外の属性に対しGroup-by操作を施した後、一つの横切り部分に対応するXの値を一つの集合要素としてまとめる操作である。関係内で集合を表すために、[ ]を用いる。Row-nest[X], Row-nest[Y]をRow-nest[X; Y]と書くこともある。Group-by操作は関数従属性、Row-nest操作は結合従属性を表現するのに適している[Kam]。

図1aに示す元の関係に対し、Group-by[学生名], Group-by[趣味]の順に非正規化を行った結果を図1b、Row-nest[趣味; 科目]を行った結果を図1cに示す。関係R(ABCDE)に対して図2aの非正規化操作を施すと、図2bのような非正規形が得られる。この例で、非正規化操作の二行目のGroup-by[B], Row-nest[D; E]はそれぞれ属性集合BCとDEに対して行われる。

3.非正規出力に関する問題

非正規出力が利用者インタフェースとして優れていることは、第一正規形による出力(図1a)と非正規形による出力(図1c)を比較すれば明らかである。本稿では、非正規出力を実現する方法として、既存のデータベースシステムを用い、データベースシステムからの正規形出力に非正規化操作を施すことにより非正規形を作る方法を用いる。この方法によれば、データベースシステムを新たに作り直すことなく非正規出力を実現できる。そのために解決すべき問題を、大きく二つにわけて考える。

学生名	趣味	科目
前田	野球	英語
前田	野球	法学
前田	ドライブ	英語
前田	ドライブ	法学
西村	映画	言語学
西村	映画	法学

図1a. 元の関係(第一正規形)

学生名	趣味	科目
前田	野球	英語
		法学
	ドライブ	英語
		法学
西村	映画	言語学
		法学

図1b. Group-by[学生名],
Group-by[趣味]

学生名	趣味	科目
前田	野球 ドライブ	英語 法学
西村	映画	言語学 法学

図1c. Row-nest[趣味; 科目]

Row-nest[BC; DE]
Group-by[B], Row-nest[D; E]

(a)非正規化操作

A	B	C	D	E

(b)非正規出力

図2. 非正規化操作と出力形式

(1)一つの第一正規形から複数の非正規形を作ることができるので、それらの非正規形の中から最適な非正規形をどのように選ぶか。

(2)作るべき非正規形の形式を決めた後、非正規形を生成するための計算量を減らすにはどうすればよいか。

例えば、図1aの第一正規形を非正規形で表現すると図1bや図1cなどのようになり、一般に数種類の表現の仕方があるので、その中から最も優れたものを選ばなければならない。これが(1)の問題である。

非正規出力を行う試みは以前から行われていた。それらの試みの大半では、非正規形の形式を利用者が自由に決定でき、データベースシステム内部の状態は考慮しない方法が用いられていた。この方法は、利用者が自由に形式を決めることができるので、利用者の意図した通りの出力形式になる利点がある。またこの方法を用いるとシステムの構築が容易になると思われる。しかし、出力形式に対する利用者の意図が必ずしも最適であるとは限らないので、出力の従属性を誤って理解しているために出力の冗長度が高くなる場合や、よりよい形式に利用者が気付いていない場合も考えられる。

そこで、基底関係が満足している意味制約と利用者の質問から、出力が満足すべき意味制約を求め、その意味制約から最適な出力形式を決める方法が提案された[KTT]。この方法によれば、出力が満足する意味制約を形式に反映しているので、出力の冗長度は低い。しかし、基底関係の意味制約と質問だけから出力形式を決めるので、必ずしも利用者の意図した形式になるとは限らないという問題がある。

(2)の問題については従来、議論されていなかった。一般に考えられる非正規形生成方法は、次のようなものであると考えられる。(図5,6)

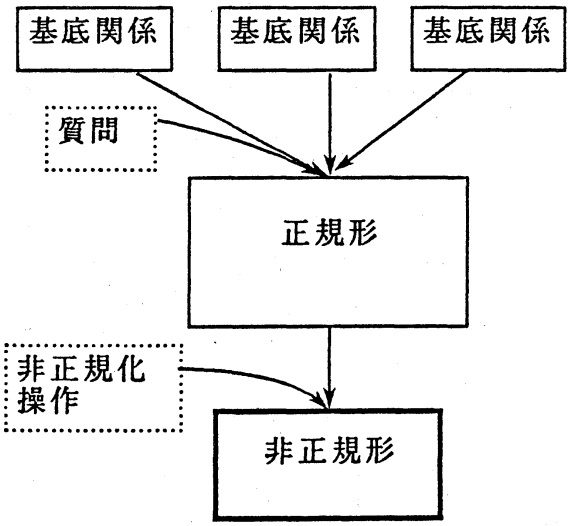

図5.従来の非正規形生成法

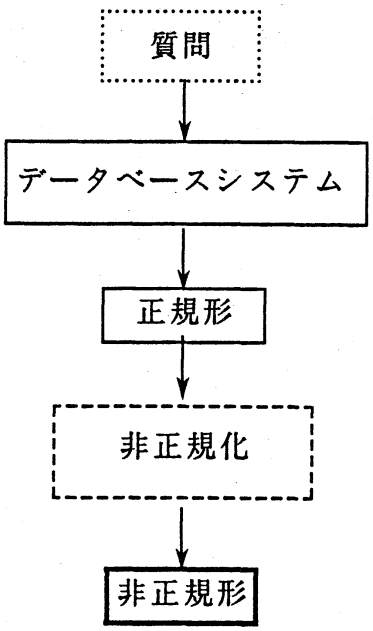

図6.従来の非正規出力システム

この方法は、データベースシステムに対して利用者の質問を直接与え、データベースシステム内で質問の実行結果である第一正規形を導出し、それに非正規化操作を施すことにより非正規形を生成するものである。しかし、この方法では導出した第一正規形の冗長度が高く、導出、非正規化にコストがかかり、効率はよくな

い。そこで、導出する正規形を従属性に基づいて複数個にすることにより、正規形の冗長度を下げ、導出、非正規化コストを下げることを考える。(図7)

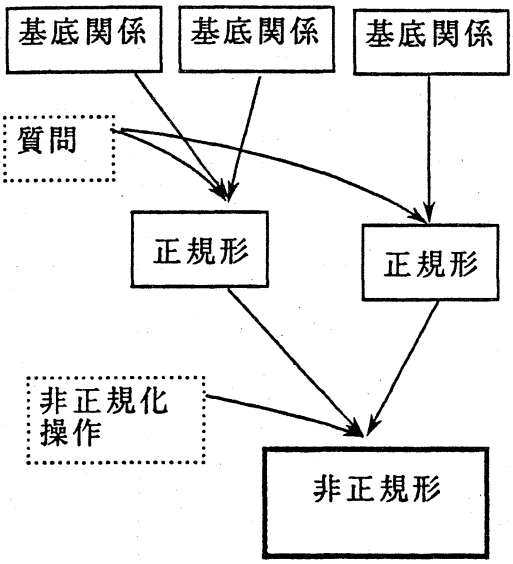

図7. 質問分解を用いた
非正規形生成法

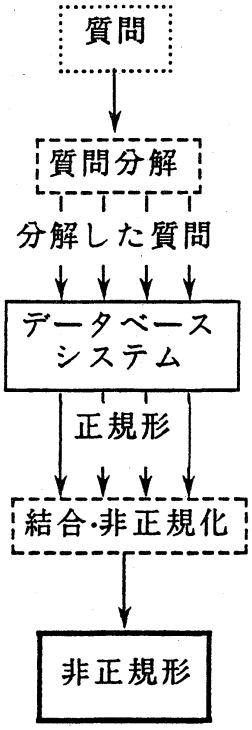

図8. 質問分解を用いた
非正規出力システム

関係データベースシステムにおいて、このように出力を複数にするには、意味制約に基づいて利用者の質問を分解して求めたい結果に対する部分的な質問を作り、それを実行すればよい。(図8)

4.質問分解法

3節で、意味制約に基づき質問を分解する方法を紹介した。この方法の場合、データベースシステムからの出力を非正規化を行うシステム側で結合しなければならない。非正規化を行う前に結合を行うと質問を分解した意味がなくなるので、まずデータベースシステムからの出力を非正規化しておき、次にそれぞれの非正規形を結合するようにする。非正規形の結合は正規形の場合とは少し異なることに注意する必要がある。

例として関係R_1(学生名, 趣味), R_2(学生名, 科目) (図9a, b)を学生名で結合する場合について考えてみる。

(1) 結合する属性ですべての関係が非正規化してある場合(R_1, R_2共にGroup-by[学生名]を行った場合)(図10)

学生名	趣味
前田	野球
前田	ドライブ
西村	映画

図9a. 基底関係R_1

学生名	科目
前田	英語
前田	法学
西村	言語学
西村	法学

図9b. 基底関係R_2

R_1

学生名	趣味
前田	野球
	ドライブ
西村	映画

図10a.

R_2

学生名	科目
前田	英語
	法学
西村	言語学
	法学

図10b.

学生名	趣味	科目
前田	野球 ドライブ	英語 法学
西村	映画	言語学 法学

図10c. Row-nest[趣味;科目]

R1

趣味	学生名
野球	前田
ドライブ	前田
映画	西村

図11a.

この場合、属性「趣味」と「科目」の値は、同じ「学生名」の値に対して直積となるので、二つの非正規形の結合は直積となる値を集合値として扱うことにより、この結合は組数がそれぞれR1, R2の異なる「学生名」の数の関係の結合と等価になり、正規形の状態で結合することに比べると結合操作のための計算量は非常に少なくなる。

(2) 結合する属性で非正規化していない関係が一つだけある場合(R_1に対してGroup-by[趣味」,R_2に対してGroup-by[学生名]を行った場合)(図11)

この場合は、R_1の「趣味」の各値ごとに横切りしたR_1とR_2を結合することになり、(1)の場合よりも結合のための計算量は増加するが、正規形の状態で結合することに比べると容易である。

R_2

学生名	科目
前田	英語
	法学
西村	言語学
	法学

図11b.

趣味	学生名	科目
野球	前田	英語
		法学
ドライブ	前田	英語
		法学
映画	西村	言語学
		法学

図11c. Group-by[趣味], Group-by[学生名]

(3) 結合する属性で非正規化していない関係が複数ある場合(R_1に対してGroup-by[趣味],R_2に対してGroup-by[科目]を行った場合)(図12)

この場合、二つの関係を非正規形の状態で結合するためには属性「趣味」と「科目」のあいだの関係がわかっていなければならない。しかし、この二つの属性は別の基底関係に属しているので、二つの関係を結合しなければ属性間の関係を求めることはできない。つまり、非正規形の状態で結合するためには基底関係を正規形の状態で結合しなければならないので、非正規形の状態では結合できない。

R_1

趣味	学生名
野球	前田
ドライブ	前田
映画	西村

図12a.

R_2

科目	学生名
英語	前田
法学	前田
	西村
言語学	西村

図12b.

趣味	科目	学生名
野球	英語	前田
	法学	前田
ドライブ	英語	前田
	法学	前田
映画	言語学	西村
	法学	西村

図12c. Group-by[趣味],
Group-by[科目]

非正規形を結合できるかどうかは、非正規化によって作り出された関係の構造が結合操作で変化するかどうかによる。まず、正規形では各組の中の値は一対一対応になっている(図13a)。非正規化を行うことは、組の中の値を一対多対応にすることを意味する(b)。3通りの非正規形の結合の仕方を考えてみる(c, d, e)。点線は結合を表す。これらのうち、cとdの場合は結合後の関係に結合前の非正規化操作の影響が反映されている。cは(1)の場合、dは(2)の場合と対応する。しかし、非正規化を行った関係は必ずbのような構造になるので、eの場合は結合前の非正規化による構造を保ったまま結合を行うことはできない。(3)の場合は、eに対応する。

このように非正規形を結合する場合には、(1), (2)のように正規形の結合よりも容易になる場合と、(3)のように結合できない場合がある。したがって、質問を分解する時、結合が容易になるように分解しなければならない。もし質問を分解すると(3)の結合が生じる場合には、質問を分解せずデータベースシステム内で正規形の状態で結合するようにする必要がある。

このように、利用者の質問の中の分解できる部分を全て分解すればよいのではなく、関係をどのように非正規化して結合するか、基底関係がどのようになっているかを考慮しなければならず、そのための情報が必要となる。

5.非正規出力インタフェースの構成

本節では、以上述べてきた問題点を解決する方法を用いた非正規出力インタフェースについて述べる。

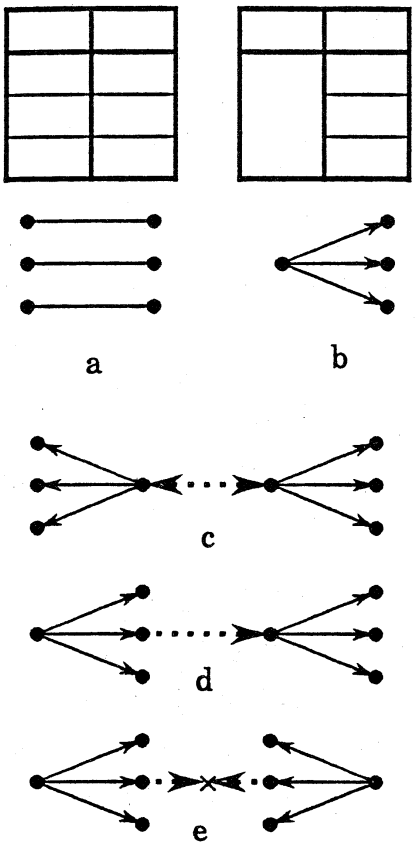

図13. 非正規形の結合の仕方

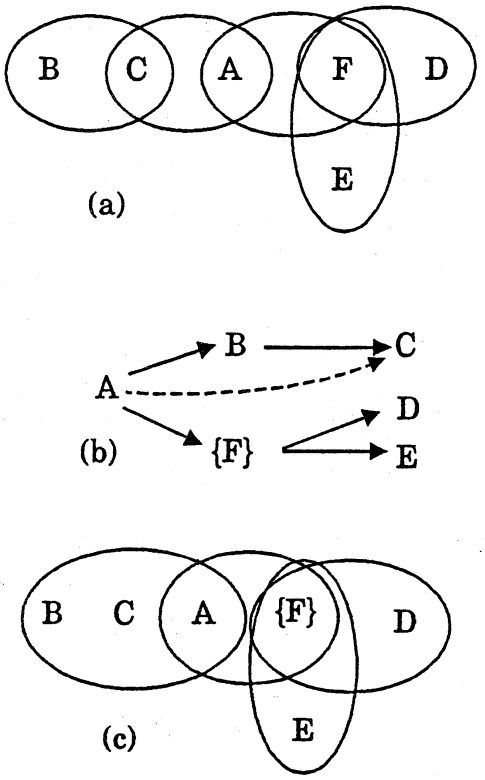

図14. 質問分解

5.1 概要

この非正規出力インタフェースは、出力形式を決める方法として、基底関係の意味制約と質問から決定する方法に基づいた方法を用いる。ただし、その形式以外は許さないのではなく、利用者が変更したい点があれば修正して最終的な出力形式を決める。

非正規形を生成する効率を向上させるために、質問を分解する手法を用いる。質問をどのように分解すればよいかを求める方法を示し、最後に最適な非正規形生成手続きを示す。

5.2 出力形式決定

利用者にわかりやすく、冗長度が低い出力を効率よく得るために、出力形式を決める部分は、基底関係の意味制約と質問から決定する方法に基づいている。これは、システムが意味制約から決定した形式と、利用者の意図する形式は大幅には異ならないと考えたためである。ただし、両者の意図が常に一致するとは限らない

ので、システムと利用者の会話により、利用者が満足できる形式を決めるようにする。その手順は基本的には以下の通りである。

(1) 利用者が質問と、出力形式に対する要求を指定する。
(2) システムは利用者の質問、データベース内部の基底関係と意味制約から出力の満足する従属性を計算する[KTT]。
(3) 利用者の要求と、(2)で計算した結果が矛盾する場合や、より冗長度が下がる形式に利用者が気付いていない場合は、利用者の要求のうち修正すべき点を知らせる。
(4) (3)をもとにして、利用者は形式を変更する。ただし、システムからの全ての通知に従う必要はない。
(5) 利用者が満足できる形式が得られるまで(3),(4)を繰り返す。この結果得られたものが、出力形式となる。

(1),(3)において、利用者は自由に出力形式を要求できる。しかし次のような場合、非正規化を行ったとしても、その効果は全く現れず無意味である。

(a) FD:A→XがありAでGroup-byする時。このような関数従属性が成立している場合、AとXの値は多対一対応となるが、Group-by[A]は、AとXの値を一対多対応にするため、結局二つの値は一対一対応となり、正規形と同じ結果となる。

(b) 結合従属性が成立していない属性集合に対して、Row-nestを行う時。結合従属性が成立していないので、Row-nestを行っても正規形と同じ結果となる。

5.3 質問分解の手順

非正規化効率を上げるためにできる限り質問分解を行うが、分解の仕方によっては逆に結合できず非正規形生成が困難になることもある。したがって、質問の分解できる部分とできない部分を明確にすることが重要である。

[定義] 非正規化グラフG_uを$G_u=(V,E)$と書き、Vは非正規化を行なう属性集合を表す節点の空でない集合、Eは非正規化を行なう順序を表す有向枝の集合を表す。節点vからwへ向かう枝はvの属性で非正規化する時だけ存在する。vから出る枝数は非正規化がGroup-byのとき1本、Row-nestのとき複数となる。

同じ基底関係に含まれる2つの属性A_1, A_2について非正規化の順がこの順で、それぞれに対応する節点v, wが$(v,w)\notin E$の場合、(v, w)をEに追加しE'とすると、次の定理が成立する。

[定理] 非正規化グラフ$G_u=(V,E')$の枝の方向を考慮しないで求めた二重連結成分が、質問を分解する属性集合である。

質問を分解する手順は、以下のようになる。

(1) 利用者が入力した質問と、5.2節で述べた方法によって決めた出力形式から、非正規化グラフを描く。同一の基底関係に含まれる属性で、非正規化の順序が隣り合っていないものの間に破線を描く。

(2) (1)で描いたグラフの二重連結成分を求める。

(3) 利用者の質問を、(2)で求めた属性集合に対する質問に分解する。

例として図14の場合を考える。(a)の基底関係のデータベースに(b)の非正規化を行なうとする。A, Cは同一基底関係に含まれるが枝で結ばれていないので、その枝(破線)を追加する。その時の出力は、図2(b)のようになる。図14(b)の二重連結成分を求めるとABC, AF, EF, DFとなり、上の定理より質問をこれらの属性集合に対する質問へ分解すればよく、(c)に示す結果が得られる。無向グラフの二重連結成分を求める問題は、深さ優先探索を用いた効率のよい解法が知られている[AHU]。

5.4 効率的な非正規形生成方式

非正規化及び結合操作回数を減らすことにより、非正規形を求める効率を上げる。以下に非正規形生成手続きを示す。

(1) 5.3節で求めた質問を実行し、その結果に対し非正規化グラフの属性で枝の順にGroup-by, Row-nestを行う。

(2) それぞれの結果に対し、出力せず非正規化だけを行う非結合属性は射影を行って取り除き、その結果を結合属性で結合する。

(3) 出力しない結合属性を射影で取り除く。その結果が、出力となる非正規形である。

一般には、結合に用いない属性は質問を実行する時点で射影を行い取り除くが、本稿の方法では非正規化に用いる属性に対しては(2)の段階まで射影を行わない。これは、意味制約の中には射影を行うと変化してしまうものがあり、そのような意味制約を(1)の非正規化によって出力結果に反映できるようにするためである。図14の例では、

(1) ABC, AF, EF, DFに対する質問を実行した結果をR_1, R_2, R_3, R_4とすれば、R_1をB, CでRow-nest、R_2をA、R_3, R_4をFでGroup-byする。

(2) Fは出力しない属性であるが結合属性なので、ここでは射影しない。R_2, R_3, R_4をFで結合する。この結合は、4節で述べた結合の仕方の(2)に相当する。次に、R_2, R_3, R_4を結合した関係と、R_1をAで結合する。この結合は、4節の(1)の結合の仕方になる。

(3) Fを射影で取り除く。

6.むすび

本稿では、利用者の要求と意味制約から決定したわかりやすく冗長度の低い形式の非正規形を、質問を分解することにより効率よく求める方法、およびその方法を用いた非正規出力インタフェースについて述べた。今後このインタフェースを実現していく予定である。実現環境としては、住友電工U-StationのUnix上で動く関係データベースシステムUnifyを用いることを想定している。

謝辞

御助言、御討論いただいた古川哲也氏、ならびに上林研究室の諸氏に感謝致します。

参考文献

[AHU] A.V. Aho, J.E. Hopcroft, J.D. Ullman, アルゴリズムの設計と解析Ⅰ, サイエンス社 (1977).

[Kam] 上林, "データ構造の意味論," 情報処理, Vol.27, No.2, pp.129-139 (1986).

[KTT] Y. Kambayashi, K. Tanaka, K. Takeda, "Synthesis of Unnormalized Relations Incorporating More Meaning," Information Sciences, Vol.29, pp. 201-247 (1983).

[SLT] N.C. Shu, V.Y. Lum, F.C. Tung and C.L. Chang, "Specification of forms processing and business procedures for office automation," IEEE Trans. Software Engineering, Vol. SE-8, No.5, pp.499-512 (1982).

[Ull] J.D. Ullman, Principles of Database Systems, 2nd Ed., Computer Science Press (1982)