Journal of the Nigerian Association of Mathematical Physics
***Volume 30,* (May, *2015*), pp257 – 262**
© J. of NAMP

Implementation of Multiple Separator Algorithm for Two Relational Operations.

***Raphael B. Adeniyi*[1], *Oludayo O. Olugbara*[2] *and I. A. Badmus*[3]**

[1]Department of Mathematics, University of Ilorin, Ilorin, Nigeria.
[2]Department of Information Technology, Durban University of Technology, South Africa.
[3]Department of Computer Science, Osun State College of Technology, Esa Oke, Nigeria..

Abstract

This paper concerns the performance of two basic relational operations namely selection and projection operations for normalizing large relational data bases. The implementation of the multiple separator algorithm for these operations is considered. For this purpose a virtual query processor for composition relation is developed and appropriate user interface for query formulation while performing the operation is suggested.

1.0 Introduction

An important class of Database Programming Problem (DPP) involves the processing of data with the following essential properties.

(i) a large volume of data is to be processed
(ii) the data has some internal connections
(iii) the data is to be made easily available to many desired users.

Codd [1] proposed relational data organisation for such problems and since then, relations have been increasingly used to solve database problems

One major problem inherent from relational approach to data management and which has received much attention within the database community is that of data redundancy or duplication. While various normalization schemes have been proposed to tackle the redundancy problem, that of duplication remains a bothering issue. Data duplication schemes involve splitting (i.e decomposition) of relations into sub-relations. Obviously decomposition is not the “silver-bullet” for solving the duplication problem. This is because decomposition itself may lead to having many sub-relations and thereby creating the problem of maintaining many relations. This also makes the altering of relations at the level of individual turples extremely difficult.

In [2] a composition scheme was proposed as an alternative to normalizing database which exhibit some form of data anomaly (especially duplication) that may be hard to remove by the classical decomposition schemes and then extended to a normalization scheme called Relation Dictionary (R-D) model with the advantage that maximum of two relations are sufficient to normalize a relation.

In Section 2, we shall consider a database anomaly called Virtual Data Duplication (VDD) and its normalization by Domain Composition (CD) scheme. Two practical database examples will be used to illustrate the VDD anomaly. Section 3 implements the basic selection and projection operations in the context of DC. A Virtual Query Processor (VQPROC) for composition relation is developed and appropriate user interface which permits a simple approach to query formulation is discussed. Finally, the paper is concluded in Section 4.

2.0 Virtual Data Duplication Anomaly

Suppose a group of petroleum product marketers lifted certain quantity of petrol from a depot in a day to a remote destination. The distribution record may be kept as represented in the following oil allocation relation scheme [3].

OIL ALLOCATION (Date, TruckNo, TicketNo, Volume, Product, Source, Destination) where the underlined attribute indicates the primary key of the relation. A feasible instance of the scheme is shown in Table 1.

Corresponding author: Raphael B. Adeniyi, E-mail: raphade@unilorin.edu.ng, Tel.: +2348038185943

Table 1: Oil Allocation Relation

<u>Date</u>	Truck No	Ticket No	Volume	Product	Source	Destination
1/8/1997	XB350AAA	AA317557	30000	Petrol	Mosimi	Abeokuta
1/8/1997	XB721LNO	AA317554	9999	Kerosene	Mosimi	Ebute-meta
1/8/1997	LA638SL	AA317543	10001	Kerosene	Mosimi	Ogijo
2/8/1997	XA342SGM	AA317559	33000	Petrol	Mosimi	MinnaS/S
2/8/1997	XB721LND	AA317544	10000	Kerosene	Mosimi	Abeokuta
2/8/1997	AS302EKY	AA317558	10000	Kerosene	Mosimi	Ita-Osin
2/8/1997	XC455JJJ	AA317558	10000	Kerosene	Mosimi	Pedler
2/8/1997	XA22JFF	AA317562	10000	Kerosene	Mosimi	Pedler
4/8/1997	XA39SGM	AA317561	30000	Petrol	Mosimi	Ijebu-Ode
4/8/1997	XB721LND	AA317567	10000	Kerosene	Mosimi	Ijebu-Ode
4/8/1997	XA591LND	AA317566	10000	Kerosene	Mosimi	Pedler
4/8/1997	XA502AKN	AA220747	30000	Petrol	Ore	Akure
4/8/1997	AQ624GGE	AA316363	33000	Petrol	Ejigbo	Oshodi
4/8/1997	AA930AGG	AA316360	33000	Petrol	Satellite	Kaduna b/F
5/8/1997	XA822JJJ	AA316365	30000	Petrol	Satellite	Akowonjo S/S

Obviously, the attributes: Product, Source and Destination are finitely small sets. Assuming we intend to further collect a larger instance of the relation for a period of one year or more, then the database grows erroneously. Many data from the small domains (i.e Product, Source and Destination) will have to be repeated in the relation. The data will therefore be duplicated and hence redundant. We call this duplication redundant because it is removable. A duplication of this nature is referred to as virtual data duplication (VDD). The use of duplicate or redundant data is widely common that no one is paying a serious attention to its implication. As another example of VDD consider the students global registration data pool represented by the single relation. REGISTRATION (MatricNo, Surname, OtherNames, Sex, Age, EntryMode, StateOrigin) with an instance shown in Table 2.

Table 2: Registration Relation

MatricNo	Surname	Other Names	Sex	Age	Entry Mode	State Origin
93/011483	Solomon	Femi Dada	M	18	UME	Lagos
93/011540	Talabi	Mji Yinka	F	18	DIECT	Oyo
93/011490	Mohammed	Aliu Jiya	M	21	UME	Sokoto
93/011500	Suleiman	Kaida Kudu	M	20	UME	Niger
93/012504	Stephen	Olloers	M	22	TRANSFER	Kogi
94/014460	Okechukwu	Phillips	M	18	UME	Imo
93/013561	Matthew	Kuye	M	18	UME	Ogun
93/013477	Yusuf	Kudu Isah	M	19	UME	Plateau
93/013474	Tsebo	Maitama	M	18	UME	Niger
93/012581	Womadi	Goodnews	F	17	UME	Rivers

The REGISTRATION relation contains an "acute" VDD anomaly because the domains: Sex, Age, EntryMode and StateOrigin are absolutely small in sizes. The next section discusses the normalization scheme appropriate for removing this type of anomaly.

2.1 Normalization by Domain Composition

The basic idea underlying the concept of normalization of large relational databases (which exhibit VDD anomaly) by Domain Composition Scheme is that two or more domains of a given relation can be combined into a single domain which is usually of integer type. A model that allows the composition of domains of different types is the Relation Dictionary (R-D) model [2], which is an extension of the DC scheme to heterogeneous domains relation.

By applying the DC scheme to normalise the relations in Tables 1 and 2 gives their normalised (i.e composition relation) copies respectively as:

NO OIL ALLOCATION (Date, TruckNo. <u>TicketNo</u>, Volume, M)

REGISTRATION (<u>MatricNo</u>, Surname, OtherNames, M)

The three sets: Product, Source and Destination for example have combined into one set called the mixture M. The concept of domain transformation [2] may be used to transform the combined (or overlapped) sets from string to integer type. Once we transform the domain of a relation, we are bound to maintain two separate relations. At this level emphasis is placed on the significance of a dictionary. Every element of the transformed domain is kept in pair with its interpretation in a dictionary, which guarantees fast retrieval mechanism. This is the basic concept of the R-D model. For the oil allocation relation example, we have two relations:

OIL ALLOCATION (Date, TruckNo, TicketNo, Volume, M)

DICTIONARY (Code, Interpretation)

The meaning of a code is the transformed string associated with that code. By maintaining two separate but inter-related relations, R-D scheme is very similar to Codd's decomposition scheme [1]. The major difference is that in Codd's scheme, two or more relations may be maintained while the R-D scheme gives optimal relation to be two.

2.2 Composition Algorithms

Let A_1, A_2 …, A_n be the n domains of a given relational database. The following overlapping algorithm can be used to compose the A_i, $i = 1,2,\ldots,n$ where n is the number of domains involved in the composition.

Algorithm 2.1: Overlapping

```
    Support = $A_i$
    For i = 2 to n
    $0_{i-1}$ = Support $\oplus$ $A_{iv}$
    Support = $0_{i-1}$
    End for
    End Overlapping
```

The resultant mixture set M^d is given by

$$M^d = 0_{n=1} \qquad (2.1)$$

M^d can be written in terms of the sets involved in the composition as:

$$M^d = A_1 \text{ ă } A_{2v} \text{ ă ú ă } A_{nv}$$

The domain 0_{i-1} for each $i = 1,2,\ldots, n$ is called the $(i-1)th$ residual. 0_{i-2} is the residual of 0_{i-1} denoted rest (0_{i-1}) and is defined by

$$0_{i-1} = Res(0_{i-1}) \text{ ă } A_{iv}\,,\; i = 1,2,3,\ldots,n \qquad (2.3)$$

The residual 0_{i-2} *of* 0_{i-1} is obtained by extracting the A_{iv} from 0_{i-1} .M^d is the residual obtained by extracting the null domain from the mixture. If A_{nv} is extracted from the mixture, M^{n-1}, the resultant residual is a mixture of degree $n-2$ that is:

$$M^{n-2} = Res(M^{n-1}) \qquad (2.4)$$

For each successive extraction of domain we have:

$$Res(M^{n-1}) = M^{n-j-1}, \quad j = 1,2,\ldots, n-1 \qquad (2.5)$$

For well composed domains, the zero residual 0_o is the residual of 0_1 given by:

$$0_o = A_1 \qquad (2.6)$$

The degree *d*, of the mixture is the number of domains that combined with a single domain called "super support" to form the mixture. If two domains A_1 and A_2 are composed such that A_1 is overlapped with A_2, then A_2 is the support of A_1 and A_1 is the base of A_2. A_{iv} is the virtual set of A_i. To obtain A_{iv} we need to generate α_i the witness of A_i and use this value to enlarge the set A_i. The algorithm which produces $\sim_i$ together with the tops and bottoms of A_i is the Gen WitBot reproduced here as follows:

Algorithm 2.2: Gen WitBot

```
    $\sim_i = ¥ = 1$
    $\perp 0_o = \perp A_1, \top 0_o = \top A_1$
    For i = 2 to n
    $\sim_i = \top 0_{i-2} + ¥$
    $\perp 0_{i-1} = \sim_i \perp A_i + 0_{i-2}$
    $\top 0_{i-1} = \sim_i \top A_i + \top 0_{i-2}$
    End Gen WitBot.
```

Algorithms 2.1 and 2.2 constitute the composition algorithms and they can be rapidly used to compose n-th domains. Application of this algorithm to the oil relation creates the R-D model for the relation. A typical instance of the model is shown in Table 3.

Table 3: R-D Model for Oil Allocation

(a) NO OIL ALLOCATION

Date	TruckNo	TicketNo	Volume	Mixture M
1/8/1997	XB350AAA	AA317557	30000	66
1/8/1997	XB721LNO	AA317554	9999	122
1/8/1997	LA638SL	AA317543	10001	123
2/8/1997	XA342SGM	AA317559	33000	69
2/8/1997	XB721LND	AA317544	10000	121
2/8/1997	AS302EKY	AA317558	10000	125
2/8/1997	XC455JJJ	AA317558	10000	126
2/8/1997	XA22JFF	AA317562	10000	126
4/8/1997	XA39SGM	AA317561	30000	72
4/8/1997	XB721LND	AA317567	10000	127
4/8/1997	XA591LND	AA317566	10000	126
4/8/1997	XA502AKN	AA220747	30000	84
4/8/1997	AQ624GGE	AA316363	33000	96
4/8/1997	AA930AGG	AA316360	33000	108
5/8/1997	XA822JJJ	AA316365	30000	109

(b) DICTIONARY

Code	Interpretation
0	Abeokuta
1	Ebute-meta
2	Ogijo
3	MinnaS/S
4	Ita-Osin
5	Pedler
6	Ijebu-Ode
7	Akure
8	Oshodi
9	Kaduna B/F
10	Akowonjo S/S
11	Mosimi
22	Ore
33	Ejigbo
44	Satellite
55	Petrol
110	Kerosene

3.0 Basic Relational Operations Applied To Composition Relation

The relational data model supports operations on relations whose results are themselves relations. These operations can be combined using the relation algebra. There are three fundamental relational operations [4]: selection, projection and join. Selection creates a subset of all tuples in a given relation by selecting out certain turples which satisfied some given conditions. One typical criterion for selection process to progress is that one or more domains have a specific value called the "selection key" which is commonly referred to as primary key. Projection creates a subset of the columns of a relation. In the resulting relation projected out, it is possible to have duplicate records which may be eliminated. Join and its variant, natural join combine two relations to produce another relations. Only the first two operations are considered in this paper.

3.1 Query Language for Composition Relation

There are three families of Data Manipulation Languages (DML): relational algebra, relational calculus and query languages. Relational algebra includes the classical operations of mathematical set theory (union, intersection and difference) as well as the relational operations select, intersection and join). Relational calculus allows a new relation to be created in terms of existing ones. The calculus allows a result of query to be defined in terms of relations in the database. Query languages are intended to exploit the facilities of the display screen.

This section describes query language specification for composition relations. To make the specification compact, we adopted the Structured Query Language (SQL) style of syntax [5], a language developed by IBM and currently being used by a large number of commercial database systems.

3.1.1 Selection: ±$_{\text{condition}}$ (R)

<u>Select</u> A$_1$ A$_2$, …, A$_m$ <u>in</u> mixture
<u>from</u> R
<u>where</u> condition
<u>giving</u> S

$A_1, A_2, \ldots, A_m$ are attributes of *R*. The result is a relation *S* whose turples are those of *R* with only the attribute $A_1, A_2, \ldots, A_m$ in the mixture set.

3.1.2 Projection: ®(R)

<u>Project</u> A$_1$ A$_2$, …, A$_k$ <u>in</u> mixture $A_{k+1}, \ldots, A_m$
<u>from</u> R
<u>giving</u> S

The result of the operation on R is a relation S whose turples are those of *R* with $A_1, A_2, \ldots, A_k$ in mixture set and $A_{k+1}, A_{k+2}, \ldots, A_m$ not in the mixture. Then every language keywords are underlined. The purpose of the "in" operation is deferred until Section 3.2.

3.2 Using Multiple Separator Algorithm to Write Query

Suppose $A_1, A_2, \ldots, A_n$ are the n-th sets each with witness $\sim_1, \sim_2, \ldots, \sim_n$ respectively. Let $M = m_1, m_2, \ldots, m_k$ be a collection of mixture m_i, $i = 1,2, \ldots, k,\ k, n$ ČZ$^+$. By Algorithms 2,1 and 2.2 each m$_i$ is formed according to the summation equation (2.7).

$$m_i = \sum_{j=1}^{n} \sim_j a_{i,j} \tag{2.7}$$

where $a_{i,j}$ is the i-th element taken from the j-th set, A_j, $i = 1,2,\ldots, r$, r is the cardinality of the relation having $A_1, A_2,\ldots, A_n$ as domains. This is the basic concept for the construction of the "Multiple Adder" function. The record insertion operation is developed by implementing equation (2.7). To separate each element $a_{i,j}$ from the mixture m_i, we employ the multiple separator algorithm. The algorithm is implemented in our Virtual Query Processor (VQPROC), listing 3.1 via the "in" operator. The processor is used to implement query with respect to the selection and projection operations. The "in" operator is used to inform the processor which selected or projected attributes are in the mixture set. Thus, the operator involves the multiple separator algorithm. Listing 3.1 is the grammar of a simple virtual query processor for query formulation in composition relation.

Listing 3.1: Virtual Query Processor

```
⟨VQPROC⟩::= ⟨Sele_Stat⟩|⟨Proj_Stat⟩|⟨Sele_Stat⟩⟨VQPROC⟩|
        ⟨Proj_Stat⟩⟨VQPROC⟩
⟨Sele_Stat⟩::=Select⟨Var_Com⟩ from ⟨Rel_Name⟩ where
        ⟨Cond_Stat⟩ giving ⟨Rel_Name⟩
⟨Proj_Stat⟩::=project ⟨Var_Com⟩ from ⟨Rel_Name⟩ giving ⟨Rel_Name⟩
⟨Var_Com⟩::=⟨Fld_List⟩|⟨Fld_List⟩ in ⟨Fld_List⟩|
        ⟨Fld_List⟩ in ⟨Fld_List⟩ : ⟨Fld_List⟩
⟨Cond_Stat⟩::=⟨Variable⟩⟨R_Oper⟩⟨Digit⟩|
        ⟨Variable ⟩⟨R_Oper⟩ ⟨Digit⟩⟨L_Oper⟩⟨Cond_Stat⟩
⟨Fld_List⟩::=⟨Variable⟩|⟨Variable⟩: ⟨Fld_List⟩
⟨Rel_Name⟩::=≤ | ≥ | = | ≠ | ≤ | ≥
⟨L_Oper⟩::= or | and
```

In Listing 3.1 variable consists of relation or field name which is a valid identifier and Digit is a number. The language terminal words are underlined.

3.3 User Interface for Query Formulation

The performance of selection and projection operation on a DC scheme requires hat an appropriate user interface be developed. This would present the user with a convenient means of interacting with the main algorithm without having to learn the rigour of the command syntax. The design of a user interface may be affected by three major considerations namely: the user's need and background, the information retrieval system and the database. For operations on composition relations, dialogue template approach to user interaction has been adopted. In this approach, direct user interaction is used to query the database. First, the database is identified, then scheme elements (i.e attributes) are displayed for selection and projection. Those elements which are relevant to the query are selected while undesired ones ignored. Suppose we select a mixed attribute then we would need also to identify the desired attribute from the mixture.

The display dialogue windows (Fig. 1) illustrates the screen layout of the VQPROC. The windows contain three list boxes: DATABASE, ATTRIBUTE and MIXTURE, an edit control box, CONDITION, a viewpoint and four pushdown buttons:

Perform, Remove, View and Exit. The content of the ATTRIBUTE box depends solely on the choice in the DATABASE box. The DATABASE box shows all the relations in the application including those dynamically created from the existing ones. The attributes of each relation is displayed in the ATTRIBUTE box. If an attribute contains several composed attributes then the MIXTURE box is filled with the attributes as soon as the composite attribute is selected. The single line "CONDITION" edit box is used to accept the condition statement for the selection operation. The view associated with the selected or projected relation is presented to the user through the viewpoint by the view command. A database can be removed or added to the application by using the Remove command which serves dual purposes. When the user select this command, VQPROC prompts him to enter the database name to create or remove. On selecting the "OK" button the processor searches for the file existence. If the file exists it is remove, otherwise a blank relation is created and added to the DATABASE list. The blank relation is to be later defined by selection or protection operation. The Perform command is used to initiate the query while the Exit command terminates it. For example, when a database file is opened (i.e selected by double clicking) the attributes to project out are selected from the ATTRIBUTE list box. The query commences as soon as the Perform is clicked. The command displays a dialogue message: "Wish to initiate the query now? If you select the "Yes" button the query commences and the last database name created by the Remove command is used as the name of the new relation. The associated view is presented in the viewpoint by the View command. On the other hand, if the user selects the "No" button the previous selection is nullified for a re-fresh. For the selection operation, we enter the selection condition (i.e Volume > 20000) in the CONDITION box and the Perform command is selected to created the relation.

Each time a new relation is created, it is added to the DATABASE box so that further operations may be performed on it. For example, the "PROOIL" is the relation created by projection of some domains from the oil relation. This mechanism enables us to recursively create new relations from the existing ones. Therefore, the combination of "selection, projection" and others can be performed on a given relation.

4.0 Conclusion

Composition relations has been presented as novel mechanism for eliminating too many relations in database applications. The effect of keeping many relations is that maintenance is hard. Every time the database is updated, there is a risk of it being damaged sine it could be left in an inconsistent state. This is particularly true when data is to be modified in several places. This problem may be alleviated with composition relations because only a maximum of two relations are kept.
Furthermore, a relation may be in 3NF (see Table 2) and still faced with the acute VDD anomaly. Composition relation is a potent device for solving this type of duplication problem. The fundamental relationship between the Codd's classical decomposition scheme and the current composition scheme is that composition relations are in 1NF. They may or may not be in 3NF.

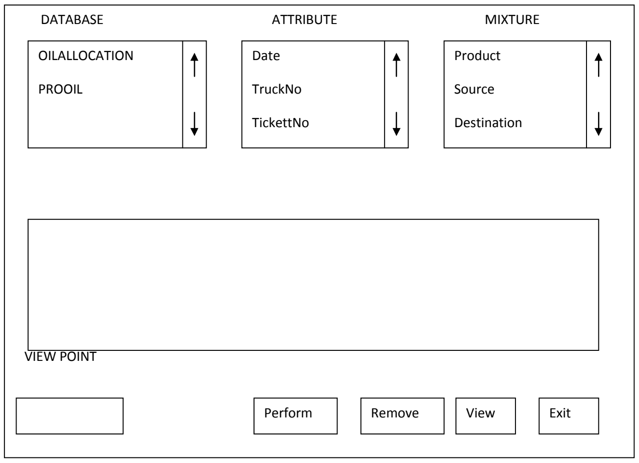

Fig. 1: VQPROC Interface

5.0 References

[1] Codd, E.F. (1970), "A Relational Model for Large Shared data Banks". Comm. ACM, 13 (6) 377-387.
[2] Adigun, M.O. and Olugbara, O.O. (1998). "Relation Dictionary Model for Large Database", Journal of Technoscience, 2 (1) 97-102.
[3] Mayne, A. and Wood, M.B. (1983), "Introducing relational database", NCC Publication, Manchester, England.
[4] Albano, A. (1994), "Database Systems", UNESCO Project. Applied Mathematics and Informatics for developing countries.
[5] Badmus, I.A. (1998), "Performance of Projection in Domain Composition Relation", Proceedings of the 14th National Conference of COAN, 9, 224-228