(10) Dr. Alaa Al-Hammami, "An Approach to the Implementation of Incomplete Databases", Ph.D., University of East Anglia.

(11) Mark L. Fussell "Tuple-Marks" V0.1 (MIf-970602] matk.fussel@chimu.com

(12) Hisham Adnan "An Approach to solve Incomplete Database Problems", 1999 Msc Thesis to University of Technology

(13) FFE Software "Should Nulls be Considered Harmful?" FEE Software, 1998.

(14) Microsoft Corporation "Windows 2000 Mc", Microsoft Corporation 2000.

(15) Hector Farcia-Molina Database System Implementation. Pretice Hall Inc 2000.

(16) Richard T. Watson "Database Management", John Wiley & Sons Inc.

(17) Patric O Neil "Database Principles, Programming, and Perfromance Academic Press 2001.

(18) Peter Rob "Database Systems", International Thomson Publishing, 1997.

(19) M. Tamer Ozsu, Patrick Valduriez "Principles of Distributed System", Prentice Hall International Inc., 1999.

(20) IEEE Computer Society "Data Engineering", December 2000 Vol. 23 No. 4

(21) Manfred A. Jeusfeld, Christoph Qiux, Matthias Jarke, "Design and Analysis of Quality Information for Data Warehouses, 2001.

(22) David McGovern "Nothing from Nothing" In Database Programming & Design, 1993.

(23) C. J. Date "Faults and Defaults" In Database Programming & Design Addison-Wesley Publishing Company, 1997.

(24) Tom Streans and Leonard Stearns "Visual Fox Pro Programming Basics" Osborne McGraw-Hill, 2000.

(25) Microsoft Corporation "Microsoft Access 2000 Help Manual", Microsoft Corporation 2000.

(26) C. J. Date and Darwen, Hugh., "The Default Values Approach to Missing Information", Relational Database Writing, Amazon 1989, 1991.

constraints as well as of data entry applications.

2. Application Specific Business-Rule proposed has the properties of practicality, simplicity, usability, minimalism, consistency, universality, and precise results.

3. Functional dependency proposed rules can be used efficiently for Null existing problems solving, practically, a designer must put all these rules determining the suitable attributes which can be treated by this approach, it gives a precise answers and results.

4. A metadata approach will be more efficient and practical when the vendors of DBMS provide the solution within the software, because it is a database function, the DBMS can handle any number of relations putting the users aware of any load, DBMS would provide facilities for generating metadata-relations and managing all the operations done previously by the user; transparently update the data among related relations after updating null values by any user at any site of the distributed database. The DBMS optimizer will be care of the performance. Metedata approach is recommended for commercially implementation.

5. Tuple-marks proposed approach is a practical useful solution for treating null values, marking them, and preventing side effects of 3VL.

6. A combined treatment for null values in distributed database is one of the powerful solutions to provide a unified treatment of different types of nulls at difficult databases.

References

(1) C. J. Date "An Introduction to Database Systems "Seventh Edition, Addison-Wesley Publishing Company, 2000

(2) Ceri Stefano & Pelagatti "Distributed Database Principles & System", McGraw-Hill Book Company, 1985.

(3) David Benyon "Information and Data Modeling", McGraw-Hill Companies 1997.

(4) Fabian Pascal "Practical Issues in Database Management" Foreword By C.J. Date Addison Wesley Publishing, 2000.

(5) Fernando Lozano "Introduction to Relational Database Design" 1999.

(6) Jeffry A. Hoffer "Modern Systems Analysis & Design" Second Edition 2000

(7) M. Tamar Ozsu Patrick Valduriez "Principles of Distributed Database Systems" Second Edition 1999.

(8) Microsoft Corporation "Visual Foxpro Programming Basics" 2001.

(9) Ramez Elmasri & Shamkant B. Navathe "Fundamentals of Database Systems" 1997.

This approach is based on two rules of solution:

First Rule (Rule-1): The extracting of unique marker va(n) is the intersection of va(n1) and va(n2).

The table below shows the results of different inputs.

va(n1)	va(n2)	n
va-mark	va-mark	va-mark (or –)
ee-null	ee-null	ee-null
ee-null	ma-null	ee-null
ee-null	li-null	li-null
ee-null	pe-null	–
ee-null	lm-null	li-null
ee-null	va-mark	va-mark
ma-null	ma-null	ma-null
ma-null	li-null	li null
ma-null	pe-ull	pe-null
ma-null	lm-null	lm-null
ma-null	va-mark	va-mark
pe-null	pe-null	pe-null
pe-null	li null	–
pe-null	lm-null	pe-null
pe-null	va-mark	–
li null	li-null	li-null (or –)
li-null	va-mark	va-mark (or –)
lm-null	va-mark	va-mark (or –)
lm-null	li-null	lm-null (or –)
lm-null	li null	lm-null (or pe-null)

Composition Rule-1

Second Rule (Rule-2):

This rule takes the composition to be the union of va(n1) and va(n2).

The possible values as va(n1) U va (n2).

All the rest results will be extracted similarly as shown in next table.

va(n1)	va(n2)	n
va-mark	va-mark	va-mark (or li-null)
ee-null	ee-null	ee-null
ee-null	ma-null	ma-null
ee-null	li-null	ee-null
ee-null	pe-null	ma-null
ee-null	lm-null	ma-null
ee-null	va-mark	ee-null
ma-null	ma-null	ma-null
ma-null	li-null	ma-null
ma-null	pe-null	ma-null
ma-null	lm-null	ma-null
ma-null	va-mark	ma-null
pe-null	pe-null	pe-null
pe-null	li-null	lm-null
pe-null	lm-null	lm-null
pe-null	va-mark	lm-null
li-null	li-null	li-null
li-null	li-null	li-null
li-null	va-mark	li-null
m-null	li-null	lm-null
lm-null	lm-null	lm-null

Composition Rule-2

5.1 Conclusions

It is a common belief that null values are one of the important features of a database application whether single data source of distributed data sources, it must be prevented.

1. Preventing Null values is obviously an important step to reduce problems. This requires an appropriate design of the database schema and integrity

4. A Combined Treatment Approach

There are many different types of null values, a practical work recommended on providing a combined treatment deals with different types of nulls, in this approach we try to do that. [4], [6], [26], [23], [20], [13].

A category of the reasons of missing data proposed which are:

1. Certainly nulls
2. Maybe null
3. Place empty null
4. Limited maybe nulls

The null attributes may be filled in with markers of any one of the following types:

Marker	Representing for
va-mark	Any element from the domain
ce-null	Certain null
ma-null	Maybe null
pe-null	Place empty null=V_S where V_S is subset from the domain of the attribute A
li-null	Limited null=V_S where V_S is subset from the domain of the attribute A
lm-null	Limited maybe null

Let:

A be a attribute
D be a domain of a given attribute
T_i be a tuple
Va be a function which takes a marker and returns a set of possible values.

The table below shows relations between markers and the function Va.

X	Va (X) Returns
va-mark	(V)
ce-null	D
ma-null	D∪{⊥}
pe-null	{⊥}
li-null	V_S
lm-null	V_S∪{⊥}

The following properties are true:

- If t_i.A is a with va(t_i.A) = {V}, then (t_i.A = v)
- If t_i.A is a ce-null, then (t_i.A ∈ Dom (A))
- If t_i.A is a li-null with va(t_i.A) = V, then (t_i.A ∈ V)
- If t_i.A is a ma-null, then (t_i.A ∈ Dom (A)) ∨ (t_i.A = ⊥)
- If t_i.A is a pe-null, then (t_i.A = ⊥)
- If t_i.A is a lm-null, with va(t_i.A) = V∪{⊥}, then (t_i.A ∈ V) ∨ (t_i.A = ⊥)

Implementation

For Combined Treatment Approach implementation, we proposed two solutions or rules for null values treatment, especially for distributed database systems. These approaches are based on null markers composition, which exists for a certain tuple in more than one site.

Assume that:

t1, t2 attribute values of site1, site2 for attribute A
n1, n2 markers of attributes
va(n1) value of attribute A according to t1
va(n2) value of attribute A according to t2

3. Metadata Approaches

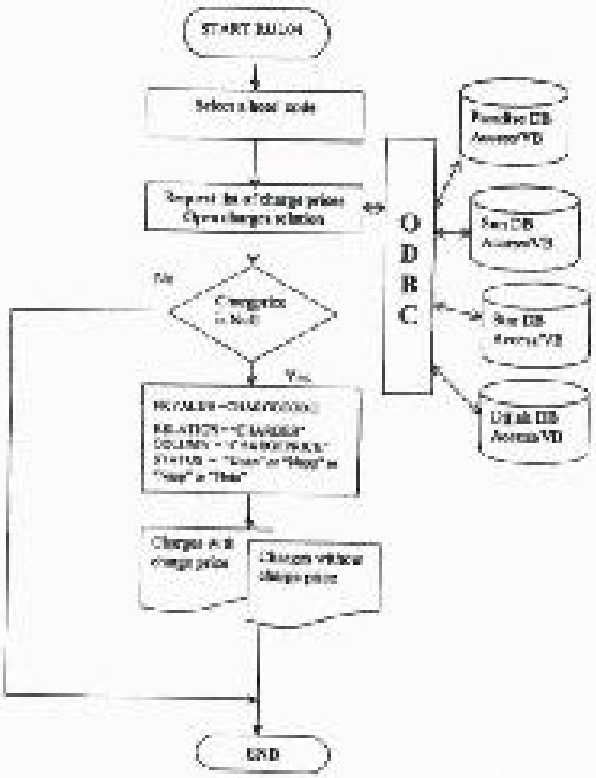
Rule04: A new relation will be created automatically by the application programs control when such null values occurrences during data insertion and manipulation. The new relations attributes will be (pkvalue, relation, column, status), where pkvalue is the primary key value and relation is the source relation name. There will be data in the metadata-relation associated with the source relation. That will be the duty of the application program or precisely triggers design for that purpose to maintain and update the metadata-relation transparently after any knowledge of the null value changed, when the unknown values become known the row updated will be triggered to delete associated row in the metadata-relation. All operations will be controlled using this approach. [4], [5], (19), (26), (23), (20), (13), (25)

PKvalue: The primary key value of the tuple contains null value.
Relation: The relation to which the null value tuple belongs
Column: The attribute name (domain) which the null value includes
Status: The status of the null value which is shown on table 3.22

For charges relation we assume that Charge_P may take Null values.

Rule04 Algorithm Steps:

1. *Open Para.mdb, Sun.mdb, Star.mdb, Dylan.dbc.*
2. *Use charges relation*
   If chargeprice not-Null: goto 9.
3. *Open metadata relation.*
4. *Pkvalue=Chargecode*
   Relation= 'charges'
   Column = 'chargeprice'
   Status = 'Unkn' or 'Napp' or 'NSup':
   goto 6.
5. *If a query request for charge code exist then: goto 8 else goto 6.*
6. *Use metadata relation and search for pkvalue=chargecode if exist then: goto 8 else: goto 10.*
7. *View table without chargeprice*
8. *View table with chargeprice*
9. *End*

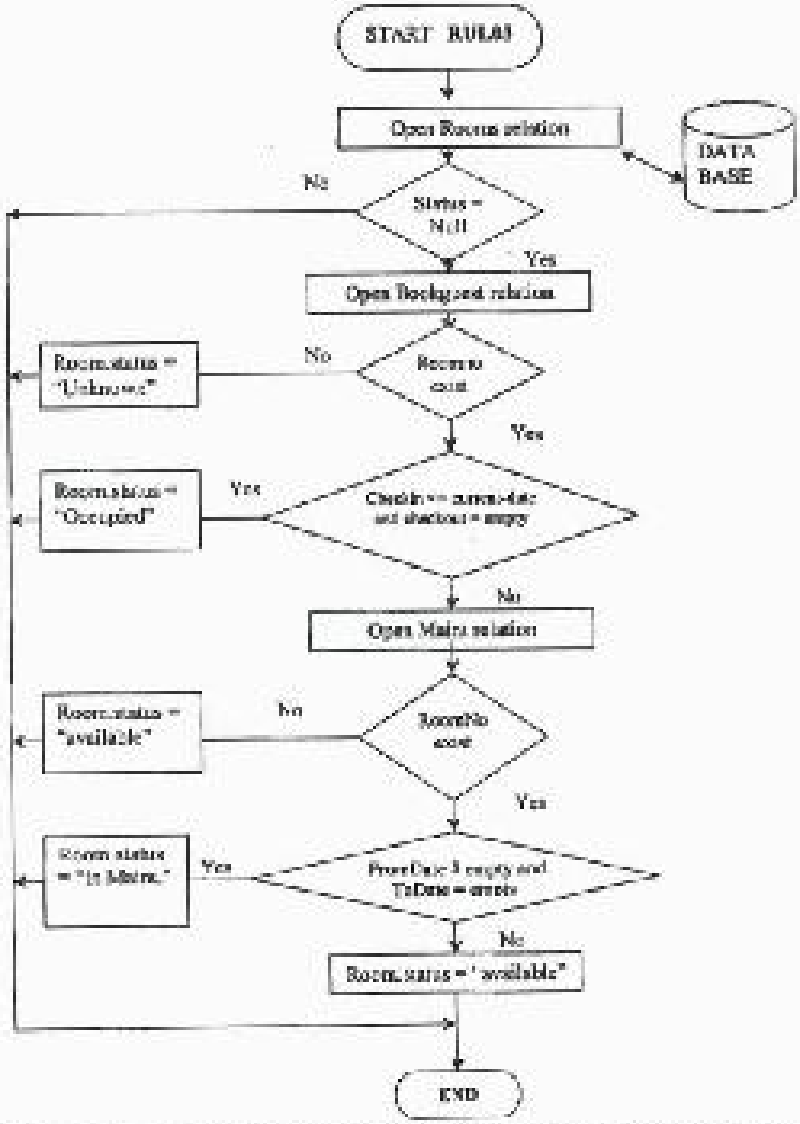


2. Functional Dependency Approach

Rule03: For treatment of room's status Null values, RULE03 is provided. (5), (19), (26), (23), (20), (18)

Rule03 Algorithm Steps:

1. *Open para.mdb*
2. *Use rooms relation*
3. *If rooms.status not=Null wroom=roomno*
4. *Use bookhest*
5. *If wroom exist: goto 7 else: goto 6*
6. *Display status= "Unknown" then: goto 11*
7. *If checkin<=current data and checkout=blank then: goto 7.a else: goto 7.b*
   7.a display status= "Occupied" then: goto 10
   7.b : goto 8
8. *Use maint*
9. *If wroom exist then: goto 9.a else: goto 9.b*
   9.a display status= "Available" then: goto 10
   9.b : goto 10
10. *If from Date not empty and empty Today then: goto 10.a else goto 10.b*
    10.a display status= "In Maintenance" then: goto 11
    10.b display status= "Available" then: goto 11
11. *End*

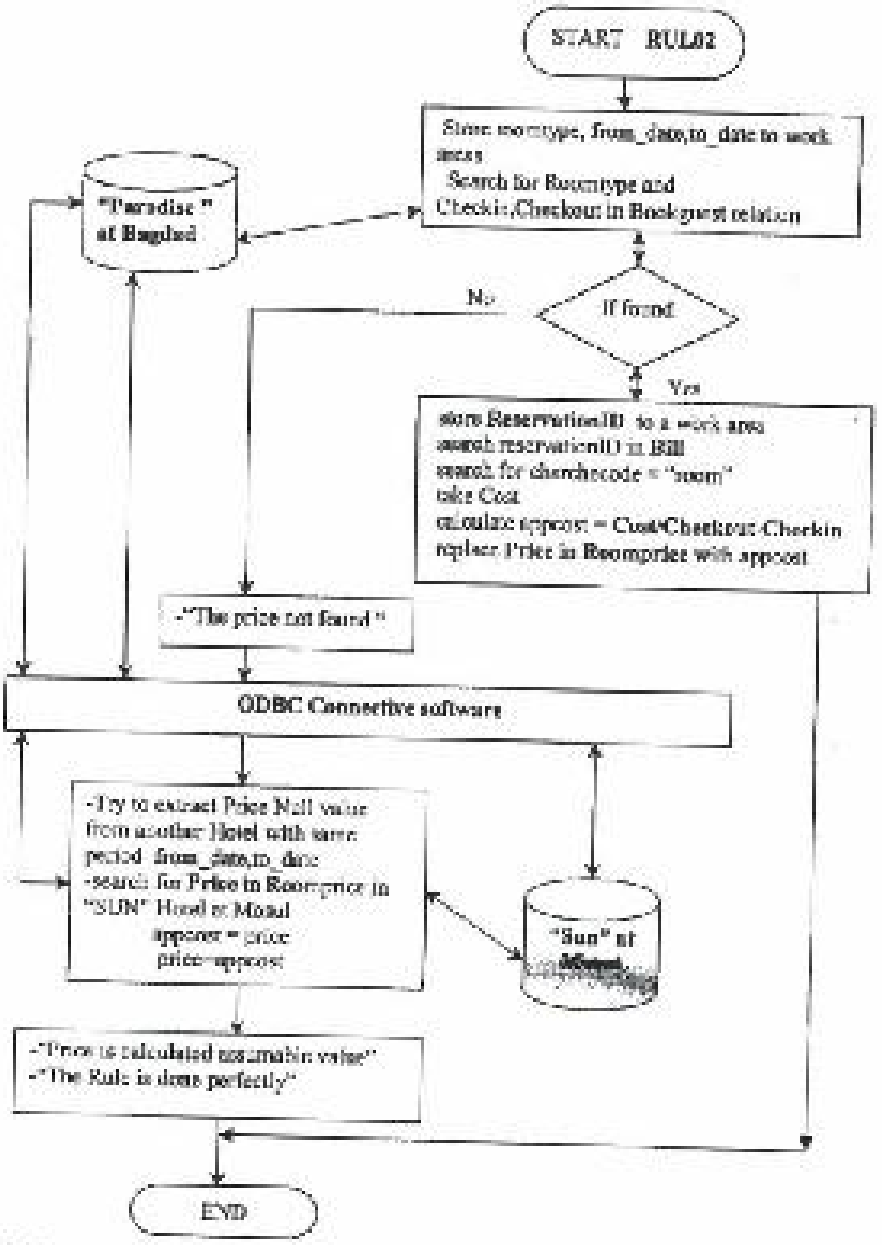


Rule03: Functional Dependency Flowchart for Null Value Treatment

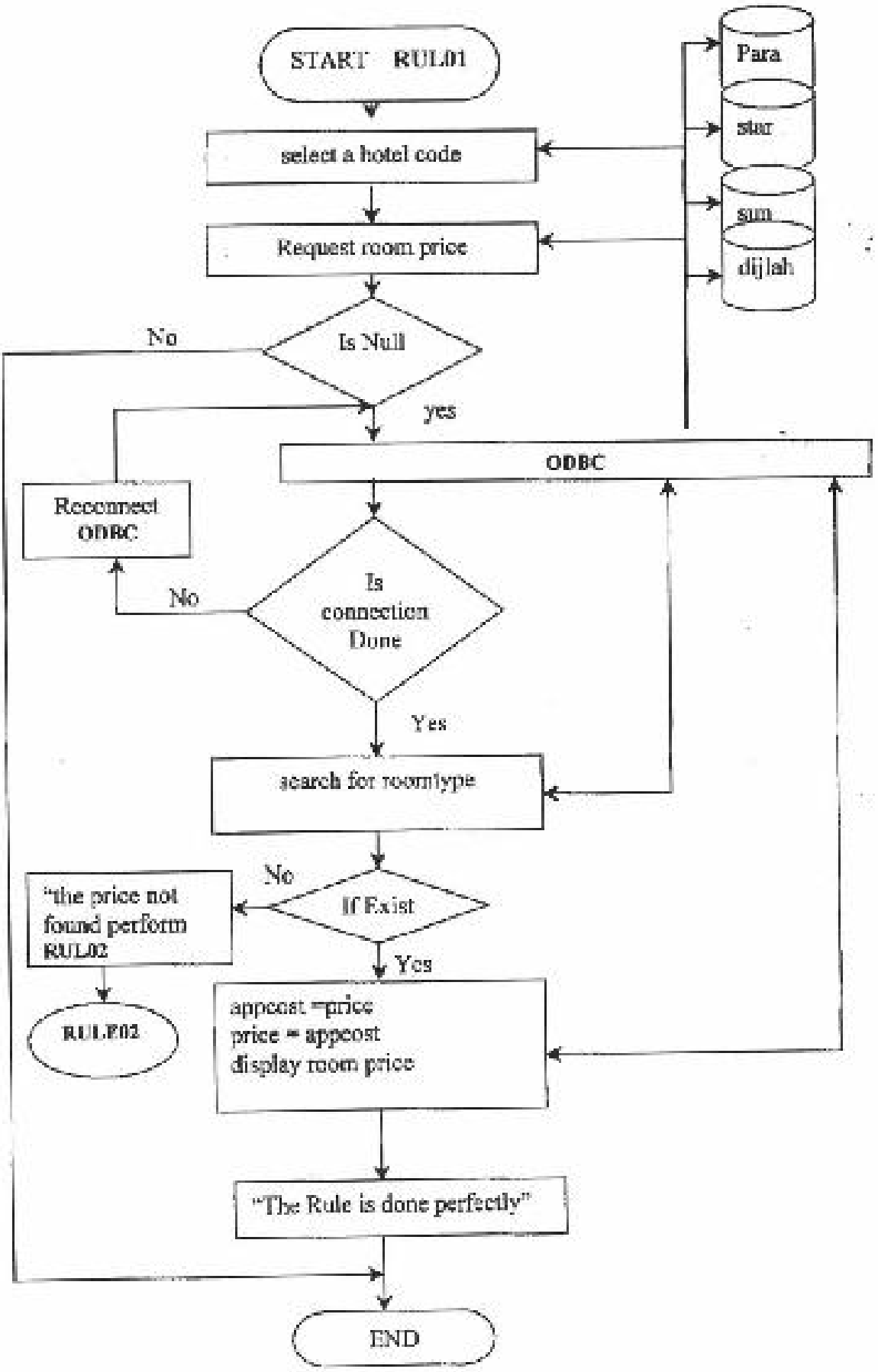
Rule02: After implementing RULE01 if there is no such exact price value extracted from a similar hotel "Dijlah", there is a need to extract a near value depending on roomtype, checkin, checkout attributes.

Rule02 Algorithm Steps

1. *Use paradise roomprice relation*
2. *Wfrom=roomprice. From_data, wto=roomprice. to_data*
3. *Use paradise bookguest relation*
4. *Check for wfrom=checkin and wto=checkout*
   a. *if exist then: goto 5*
   b. *else: goto 9*
5. *Wreserv=Reservation ID then goto 6*
6. *Use Bill relation then: go to 7*
7. *Check for ReservationID=wreserv*
   a. *if exist then : goto 7*
   b. *else display "room price not fmund then: go to 9*
8. *Appcost=cost/wto-wfrom then goto 12*
9. *Open SUN, mnh through ODBC*
10. *Use roomprice relation*
11. *Find for wfrom=roomprice. from_data, wto=roomprice.to_data*
12. *Appcost=roomprice.price*
13. *Display a message "price is calculated assumable value"*
14. *End.*

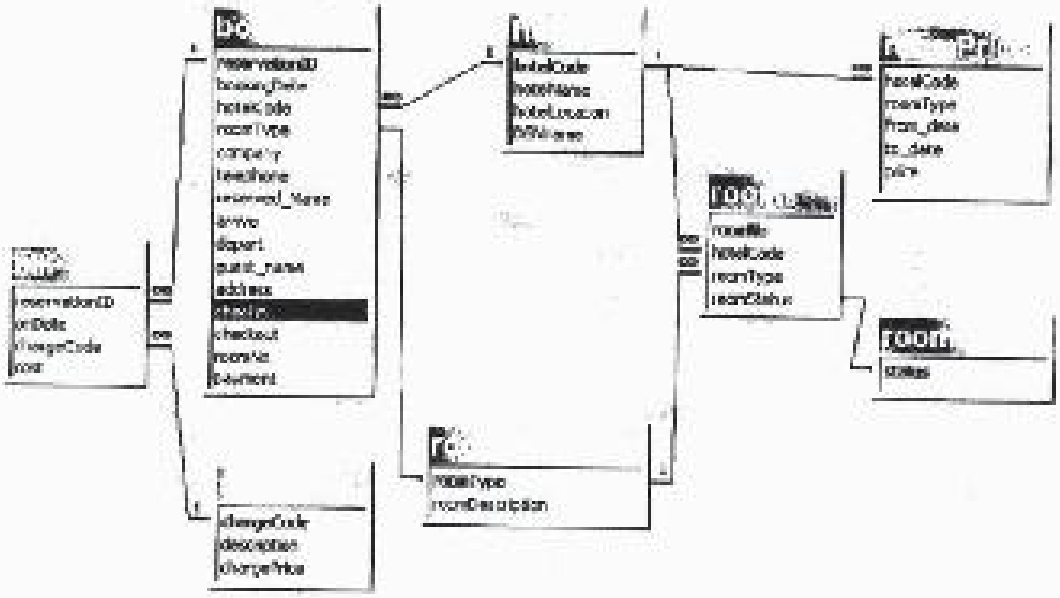


Rule02: A Business Rule Flowchart for Null Value Treatment


START RUL01
Para
star
sun
dijlah
select a hotel code
Request room price
Is Null
No
yes
ODBC
Reconnect ODBC
Is connection Done
No
Yes
search for roomtype
"the price not found perform RUL02
No
If Exist
Yes
RULF02
appcost =price
price = appcost
display room price
"The Rule is done perfectly"
END

لثلاثة فنادق أما الفندق الرابع فاستخدم (Visual FoxPro) كما واستخدمت مجموعة الدوال المعرفة ODBC لتأمين الربط بين مختلف قواعد المعلومات. تم تقديم خمسة حلول مختلفة لمعالجة حدوث القيم اللاغية كمحاولة لتقديم حلول عامة يمكن تطبيقها عمليا في بيئة قواعد المعلومات الموزعة.

في بحثنا هذا حاولنا تقديم حلول شاملة لخدمة المستفيدين المشابهين، فلقد تم تصميم نظام قواعد معلومات موزعة ليصبح الأرضية الملائمة لتقديم وتطبيق الحلول المقترحة، أن النظام هو ((إدارة فنادق)) لشركة مفترضة تمتلك أربعة فنادق في العراق موزعة في مدن وأماكن جغرافية مختلفة: تم تصميم النظام وبرمجته باستخدام لغة البرمجة والتصميم ((Access/Visual Basic))



Hotel Database Tables and Relationship

Proposed Approaches for Null Values Treatment

For approaches are proposed for treatment of Null values in distributed database applications, the summaries written below give practical use and another known approach is implemented it for nulls treatment. (1), (4), (13), (4), (5), (19), (26), (23), (20)

1. Application of Specific Business-Rules.

Rule01: For a RoomPrice relation when price is Null for example we can extract the exact value from the rule as shown below: (5), (19), (26), (23), (20), (12)

Rule01 Algorithm Steps

1. Open a hotel database
2. Use roomprice relation
3. Check for roomprice. price if exit then: goto 10 else wtype=roomprice. roomtype
4. Open dijkh. dbc through ODBC
5. If connection failure: goto 4 else
6. Use roomprice relation then
7. Search for wtype, if exist then: goto 8, else, perform RUL02
8. Appoint= roomprice. price
9. Display a message "RUL01 is done perfectly "then: goto 10
10. End.

Null Values Treatment in Distributed Databases

By

Saran Akram Abd Al-Majeed

Information Technologies Center

Abstract

There has been a great deal of discussion about null values in relational databases. The relational model was defined in 1969, and Nulls was added in 1979. Unfortunately, there is not a generally agreeable solution for null values problem.

Null is a special marker which stands for a value undefined or unknown, which means that no entry has been made, a missing value mark is not a value and not of a data type and cannot be treated as a value by Database Management System (DBMS). As we know, distributed database users are more than a single database and data will be distributed among several data sources or sites, it must be precise data, the replication is allowed there, so complex problems will appear, then there will be need for perfect practical general approaches for treatment of Nulls.

A distributed database system is designed, that is "Hotel reservation control" system, based on different data sources at four sites, each site is represented as a Hotel, for more heterogeneity different application programming languages there are five practical approaches, designed with their rules and algorithms for Null values treatment, through the distributed database sites. [1], [2], [3], [4], [5], [9]

معالجة القيم اللاغية في قواعد البيانات الموزعة

ملخص البحث

إن القيم اللاغية هي جزء من تمثيل العالم الحقيقي، وأن هناك اهتمام كبير في السنين الأخيرة من هذا الموضوع، من نقاشات وبحوث جامعية ومحاولات المعالجة واقتراح طرق لذلك، كما نعلم بأن النموذج العلائقي عرف في عام ١٩٦٩ ولفترة عشرة سنوات لم يكن هنالك مفهوم يدعى قيم لاغية (Null Values) وقد بدأ عرض وتطبيق التعريف في عام ١٩٧٩، أن كل الحلول والنقاشات حول الموضوع لم تؤدي إلى نتيجة مقبولة من ناحية التطبيق العملي.

إن القيم اللاغية تعرف للكينونة بأنها علامة أو ماركة تاشير خاصة بالأشياء أو المعرفات غير المذكورة أو المعرفة والتي تعني بأنه لا يوجد إدخال للبيانات لهذا المعرف أو الكيان، أن الإشارة هي ليست بقيمة ولا يمكن معالجتها كقيمة بواسطة أنظمة إدارة قواعد البيانات (DBMS)، أما في قواعد البيانات الموزعة فهناك خصوصية لها، فهنا الأمر يختلف من ناحية كثرة إعداد المستخدمين وتنوع مواقعهم الإدارية أو الجغرافية، إلخ. ولتنوع مصادر المعلومات وشفافيتها وإمكانية التقسيم للبيانات أو حدوث التكرار فيها، فلكل هذه الأسباب فإن إيجاد حلول لمعالجة القيم اللاغية سوف يؤدي إلى تحسين في أداء منظومات قواعد البيانات الموزعة، وزيادة الثقة بالمعلومات المخزونة فيها و الوصول إلى نتائج و أجوبة دقيقة منها لمختلف الاستفسارات والتقارير والإحصائيات.