Global Journal of Computer Science and Technology
Volume 11 Issue12 Version 1.0 July 2011
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

# Maintenance Modification Algorithms and its Implementation on object oriented data warehouse

By Dr. Mrs Pushpa Suri , Mrs. Meenakshi Sharma

*Kurukshetra University, Kurukshetra Haryana, India*

*Abstracts -* A data warehouse (DW) is a database used for reporting Paper describes Modification Algorithm and implementation on Object Oriented Data Warehousing. A Data Warehouse collects information and data from source data bases to support analytical processing of decision support functions and acts as an information provider. In initial research data warehouses focused on relational data model. In this paper concept of object oriented data warehouse is introduced modification maintenance algorithms and its implementation to maintained consistency between data warehouse and source data base.

*Keywords :* *Data warehousing, object oriented database, instance, maintenance*

*GJCST Classification :* *H.2.7, H.2.8*


*Strictly as per the compliance and regulations of:*

# Maintenance Modification Algorithms and its Implementation on object oriented data warehouse

Dr. Mrs Pushpa Suri[α], Mrs. Meenakshi Sharma[Ω]

*Abstract* - A data warehouse (DW) is a database used for reporting Paper describes Modification Algorithm and implementation on Object Oriented Data Warehousing. A Data Warehouse collects information and data from source data bases to support analytical processing of decision support functions and acts as an information provider. In initial research data warehouses focused on relational data model. In this paper concept of object oriented data warehouse is introduced modification maintenance algorithms and its implementation to maintained consistency between data warehouse and source data base.

*Keywords : Data warehousing, object oriented database, instance, maintenance*

## I. Introduction

The concept of data warehousing was first proposed by Inmon (Inmon and Kelley,1993). A data warehouse is a repository of subjectively selected and adapted operational data which can successfully answer any ad hoc, statistical, complex or analytical queries. Data warehousing technology is becoming essential for effective business intelligence, business strategy formulation and implementation in a globally competitive environment where in larger and larger amounts of data are required to be processed faster and faster for comprehension of its real meaning and impact. The term Data Warehouse was coined by Bill Inmon in 1990, which he defined in the following way: "A warehouse is a subject-oriented, integrated, time-variant and non-volatile collection of data in support of management's decision making process". Data that gives information about a particular subject instead of about a company's ongoing operations. It is integrated as data that is gathered into the data warehouse from a variety of sources and merged into a coherent whole.

Data warehouse system is time variant as all data in the data warehouse is identified with a particular time period. Data is stable in a data warehouse. More data is added but data is never removed. This enables management to gain a consistent picture of the business. (Source: "What is a Data Warehouse?" W.H. Inmon, Prism, Volume 1, Number 1, 1995). A single-subject data warehouse is typically referred to as a data mart, while data warehouses are generally enterprise in scope. Also, data warehouses can be volatile. Due to the large amount of storage required for a data warehouse, (multi-terabyte data warehouses are not uncommon), only a certain number of periods of history are kept in the warehouse. For instance, if three years of data are decided on and loaded into the warehouse, every month the oldest month will be "rolled off" the database, and the newest month added. Data warehouse contains information that is being collected from different sources and integrated into a common repository for efficient query and analysis. When the data sources are disturbed over a different location then a DW has the responsibility to collect the necessary data and save it in appropriate form. In this paper some research topics are mentioned as Maintenance [11] [12] [13] [15] [17][20] [21], consistency[6],[26],[27].

In this paper is organized as follows. The concept of object oriented data warehousing. Formal definition of class instance and object oriented data warehousing. Maintenance modification algorithms for consistency between the object oriented data warehouse. Examples are also given there to illustrate the proposed algorithms and its implementation in oracle 10g.

## II. Object Oriented Data Warehousing

In an object oriented database, each employee or class is associated with unique identifier, a set of attributes and a set of procedures. There could be no. of data types such as atomic or any other class. Object Oriented Data warehousing, like other areas of Information Technology, is a field in the midst of change. The current systems integration approach is associated with the objective of creating a centralized operational data store and Decision Support System read-only server-based application [3]. To meet this objective, it is necessary to extract, transform, and transport data from

*Author α : Associate Professor, Deptt. Of Computer Science. & Applications, Kurukshetra University, Kurukshetra Haryana, India. Email : pushpa.suri@yahoo.com*
*Author Ω : Assistant Professor Deptt of Computer Science & Engineering in H.C.T.M, Kaithal Haryana, India.*
*Email : minny_kaushik@yahoo.com*

isolated islands of information to such centralized repositories, and then to retrieve information efficiently and effectively through query and reporting tools. To perform multidimensional analysis, and to meet performance criteria, special methods and tools associated with On-Line Analytical Processing are employed. Multidimensional client, multidimensional server (Multidimensional Online Analytical Processing or Multidimensional Data Online Analytical Processing), Relational Online Analytical servers, and most recently, Vertical Technology servers, are used to help performance in the query and reporting process. Object Oriented Data Warehouse approaches are better at specifying user requirements than Systems Integration ones. In particular, systems integration approaches seem to move from process identification to data modeling without specifying the details of the identified processes. They do not employ Use Case specification and analysis to get at requirements, while this is a central aspect of Object Oriented Data Warehouse .The fourth class of specific reasons for an Object Oriented Data Warehouse approach to data warehousing is conceptual consistency with the various components of a data warehousing solution. The tools used to arrive at these solutions are increasingly object-oriented. For example, data extraction, transformation and transportation (ETT), tools from Sagent, Informatica, Carleton, ETI, VMARK and others strongly reflect the conceptual outlook of object technology. In the object oriented Data warehousing we used the classes and instances. An object type is a description of a set of object sharing the same attributes operations and relationships. Classes are implementation of types in software. So, objects are instances of classes as well as one of the types.

## III. Notation And Definition

In an object oriented data base system, we have defined certain definitions for various employees or classes. The classes can be organized according to their hierarchy. Let ID be a set of identities, A be set of attribute names be a set of data types allowed for A,TW be a set of atomic data types be a set of values and M be a set of processing methods. A set of employees in an object oriented database can be defines as follows:

*a) Definition (Class)*

A class is used to create new instances (objects) by instantiating the class. A class usually represents a noun, such as a person, place or (possibly quite abstract) thing - it is a model of a concept within a computer program. Fundamentally, it encapsulates the state and behavior of the concept it represents. It encapsulates state through data placeholders called attributes (or member variables or instance variables); it encapsulates behavior through reusable sections of code called methods.[**source : Molina G H (1995)**].

A class c is a quadruple {cid, ca, ct, cm} where cid ε ID, ca = < ca1, …… can> with cai ∈ A and i= 1 to n, ct = < ct1, …… ctn > with ctj ε T and j=1 to n, and cm ⊆M.

ID: Collection of identifies (Name) of classes
A: Collection of attributes of classes of collection ID
T: Collection of type of attributes of collection A
M: Collection of methods of classes of collection ID

***Example 1*** *:* In this example four classes, **EMPLOYEE, NAME, OFFICE**, and **DEPT** are taken. The class **EMPLOYEE** has four attributes, **EmployeeID, EmployeeName, EmployeeDept, EmployeeTitle** and one method **Counter().** In this example attribute EmployeeID is a character type, the attribute **EmployeeName, EmployeeDept** and **EmployeeTitle** is a character type. In the graph shown in figure 2 circles with shadow represents classes and circle represent in instance.

1. Class **NAME** {
First char(20), Middle char(20), Last char(20)}
2. Class **OFFICE** {
State char(20), City char(20))}
3. Class **DEPT**{
DeptId char(3), DeptName char(40), DeptOffice Office , counter() int}
4. Class **EMPLOYEE** {

EmployeeID char(20), EmployeeName Name, EmployeeDept Dept, EmployeeTitle char(10), Counter()int}

For the class Employee in this example, cid= Employee, ca= {EmployeeID, EmployeeName, EmployeeDept, EmployeeTitle }, ct= {char, Name, Deptart), and cm= {Counter()}, Let C be the set of classes defined in source database. C= { c1,c2,…….cn} where $c_i$ is a class, $1 \le i \le n$ .

*Figure 1 :* An example 1 of classes

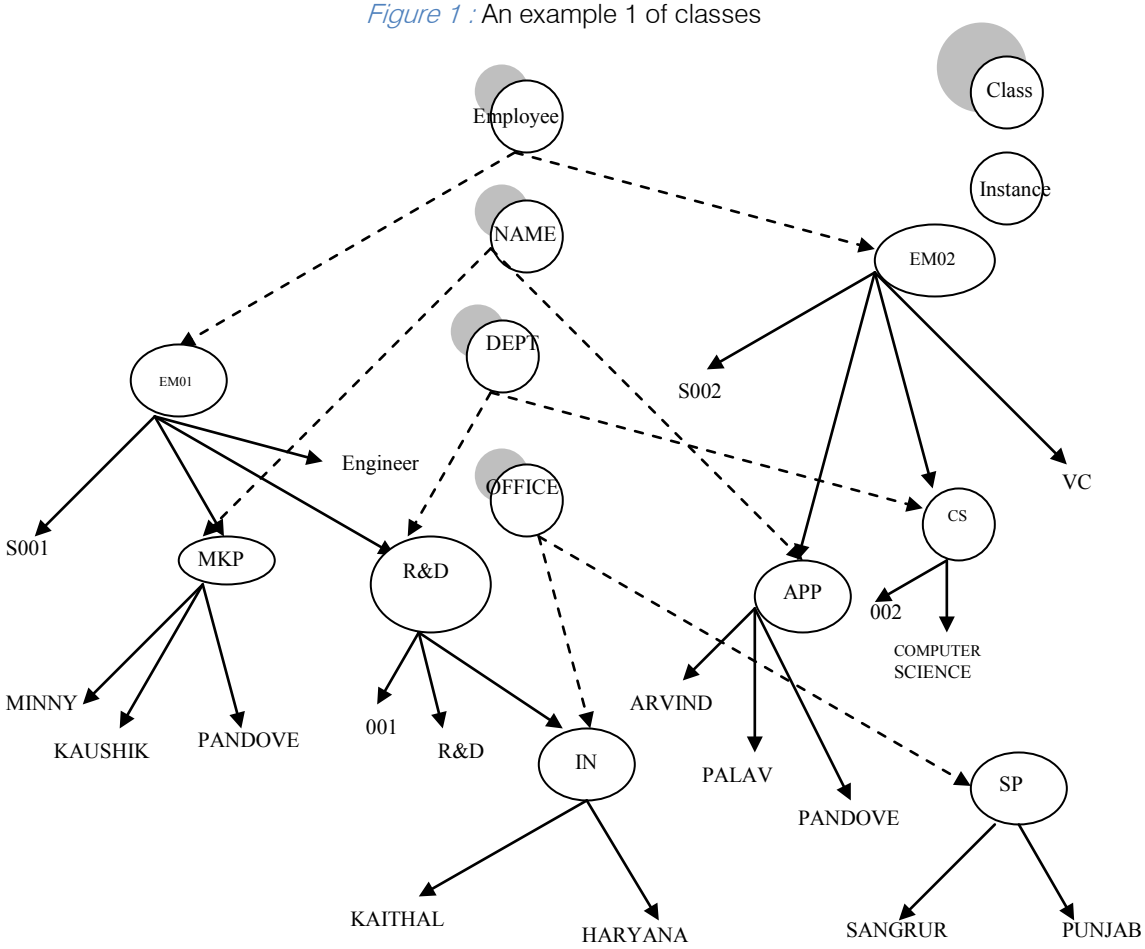


*Figure 2 :* Graphical representation of Classes

*b) Definition (Instance)*

An instance t ={tid, ta, tv,tm}is created and inherits from a certain class cid={cid, ca, ct, cm} such that tid ∈ ID, ta= ca tv= $<tv_1, tv_2, \ldots\ldots tv_n>$ with $tv_i \in$ U and $tv_i$ being of type $ct_i$ for i=1 to n, and tm= cm.

***Example 2 :*** For the example in Figure 2, assume that two instances are created by and referring to class *Dept.* One is called *R&D* with attribute values (001, R&D) the other is called *CS with* attribute values (002, Computer Science). Similarly, assume that two instances, *A1* and *B1* respectively, with attribute values (001, R&D) and (002, CS) are created by referring to the class Dept. Assume that two instances, *MKP* and *APP* respectively, with attribute values (MINNY, KAUSHIK, PANDOVE) and (ARVIND, PALAV, PANDOVE) are created by referring to the class *Name.* Also assume that two instances, *EM01* and *EM02* respectively, with attribute values (S001, MKP, AI) and (S002, APP, BI) are created by referring to the class *Employee.*

*c) Definition (Data warehouse)*

An object-oriented data warehouse W is a triple {V, VC, I}, where V is the set of view definitions, VC is a set of classes and I is the set of instances generated from the source database according to VC and V. Below, modification maintenance algorithms are proposed to maintain the consistency between an object-oriented data warehouse and its underlying source databases. They are instance insertion, instance modification alters, and instance modification update.

## IV. Instance Insertion

We have a source database, a new instance Iid is inserted into a source database. A new Msg known as transaction Msg is formed and sent from the data collector to the data warehouse for the view maintance. The proposed syntax of the transaction Msg for instance insertion as follow : MID, insert, Iid, Cid

In this Msg identifier of this transaction which is formatted automatically by data collector. Insert is denote type of Msg. Iid identifier the new instance which is inserted in a database and Cid class identifier form which this instance is inherits.

**The algorithm of maintenance for instance insertion**

**Input : -** A Data Warehouse W(V,VC, I) and an instance

lid of the class Cid is inserted into the source database.

**Output :** - A modified Data warehouse W' (V, VC, I')

**Step1 :** A source receives an instance insertion truncation Message, which is formed and sent from the data collector to the data warehouse.

**Step2** : Make the view definition to find the definition which refers to the class Cid in the From Part. View found is denoted by $V_A$.

**Step3** : If A is empty, set W'= W and exit the algo otherwise go to the next step.

**Step4** : After application of select, where operations deduce all the attributes from the view named $V_A$ and denotes it by $V_B$.

**Step5** : Request the data collector to collect the contents of $V_B$ and instance lid or alternatively from its subsequent descending instances.

**Step6** : Acknowledge the contents of $V_B$ from the data collector.

**Step7** : If contents of $V_B$ received and satisfy the conditions of view v in A. Create a new instance according to the class of the view otherwise do nothing.

**Step8 :** After step 7, new necessary instances are created and inserted into the Data warehouse. Data warehouse now modified by.**W' (V, VC, I')**

## V. Instance Modification Alter

When the attribute values of an instance tid in the source database are changed, a transaction Message is sent from the data collector to the data warehouse for view maintenance. In algorithm W is warehouse, V is View, C is the Class and I is Instance. The format of a transaction message for modifying an instance is proposed as follows:

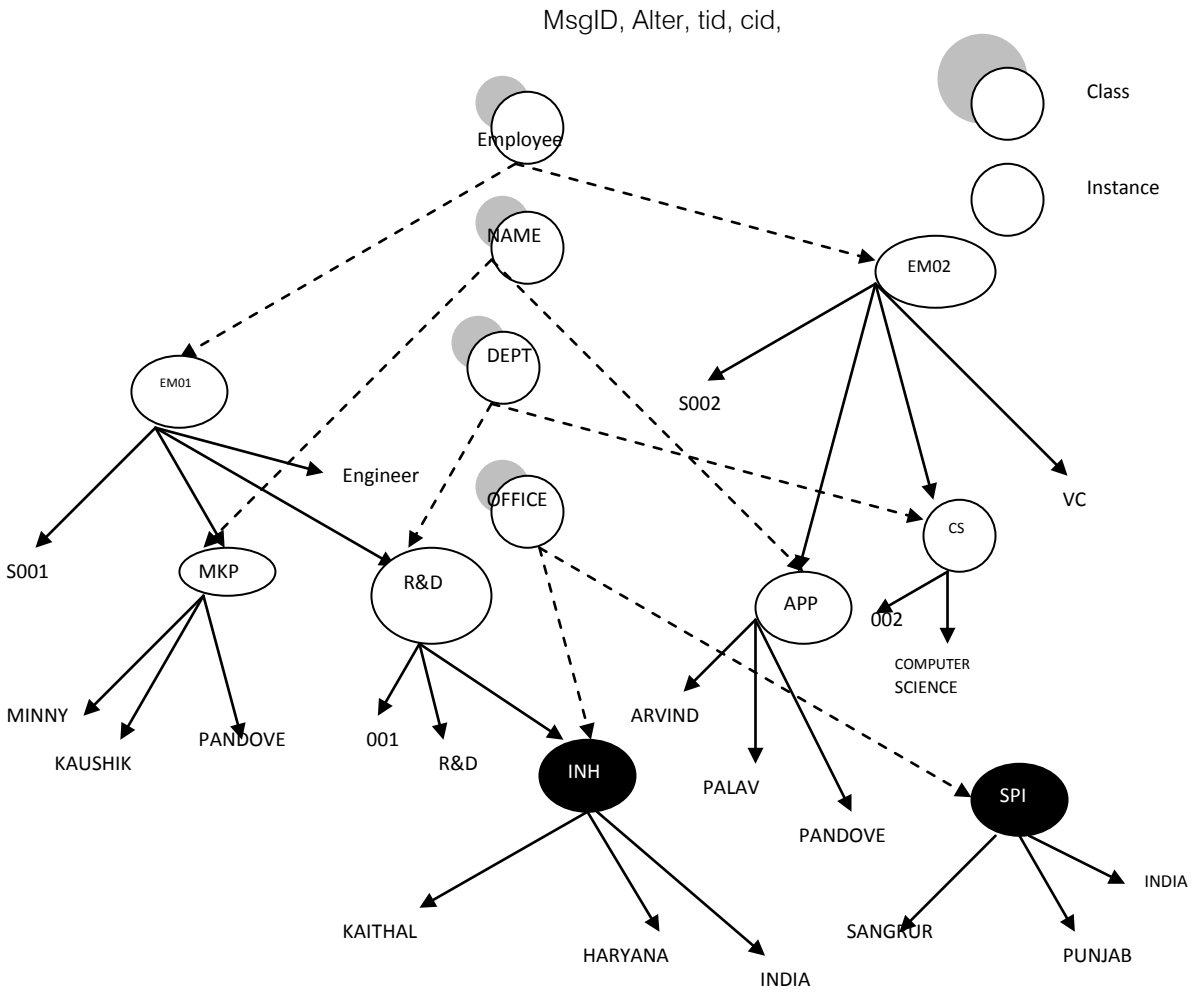


*Figure 3 :* The Graphical Representation of Instance Modification Alter for example 3

(From Figure 2 to Figure 3)

Where ui denotes the i-th attribute add to be alter and vi denotes the new attribute. Example3, assume that the attribute office in instance IN is alter from IN to INH. The data collector will detect it and send a transaction message (001, alter, office, R&D, {(office country)}) to the warehouse. The maintenance algorithm for processing the above instance modification alters is proposed as follows:

**The maintenance algorithm for instance modification**

**Alter:**

**Input:** Data warehouse W(C, V, I) and modified alter instance tid of class cid.

**Output:** A revised data warehouse W' (C, V, I')

**Step1**: An Instance of modified alter message is received from the data collector.

**Step2**: Search the data warehouse W for instance tid: If instance tid exits in W exist algorithm otherwise Go to the step No.3 and set W'=W

**Step3:** For the instance tid in warehouse alter its attribute according to the transaction message.

**Step4:** If the instance tid satisfies the condition of at least one view which refer to class cid, then keep the instance tid in I of the warehouse W otherwise remove tid from I in the warehouse W

The attribute of instance tid have been modified alter add in the data warehouse after the Step no.4.

**Example 3 :** Assume that the attribute of an instance have been added in the source database and the transaction Msg is formed as Alter type office add attribute (country char(20))cascade. This message is processed by the instance modified alter algorithm as follow:

**Step1:** Receive the transaction message alter type office add attribute (country char(20)) cascade from the data collector.

**Step2:** Since the instance A1 exists in the warehouse W then stop.

**Step3:** Alter the attribute country of the class office.

**Step4**: If A1 satisfies the condition of view country office it is kept in W.

## VI. Instance Modification Update

When the attribute values of an instance tid in the source database are changed, a transaction message is sent from the data collector to the data warehouse for view maintenance. The format of a transaction message for modifying an instance is proposed as follows:

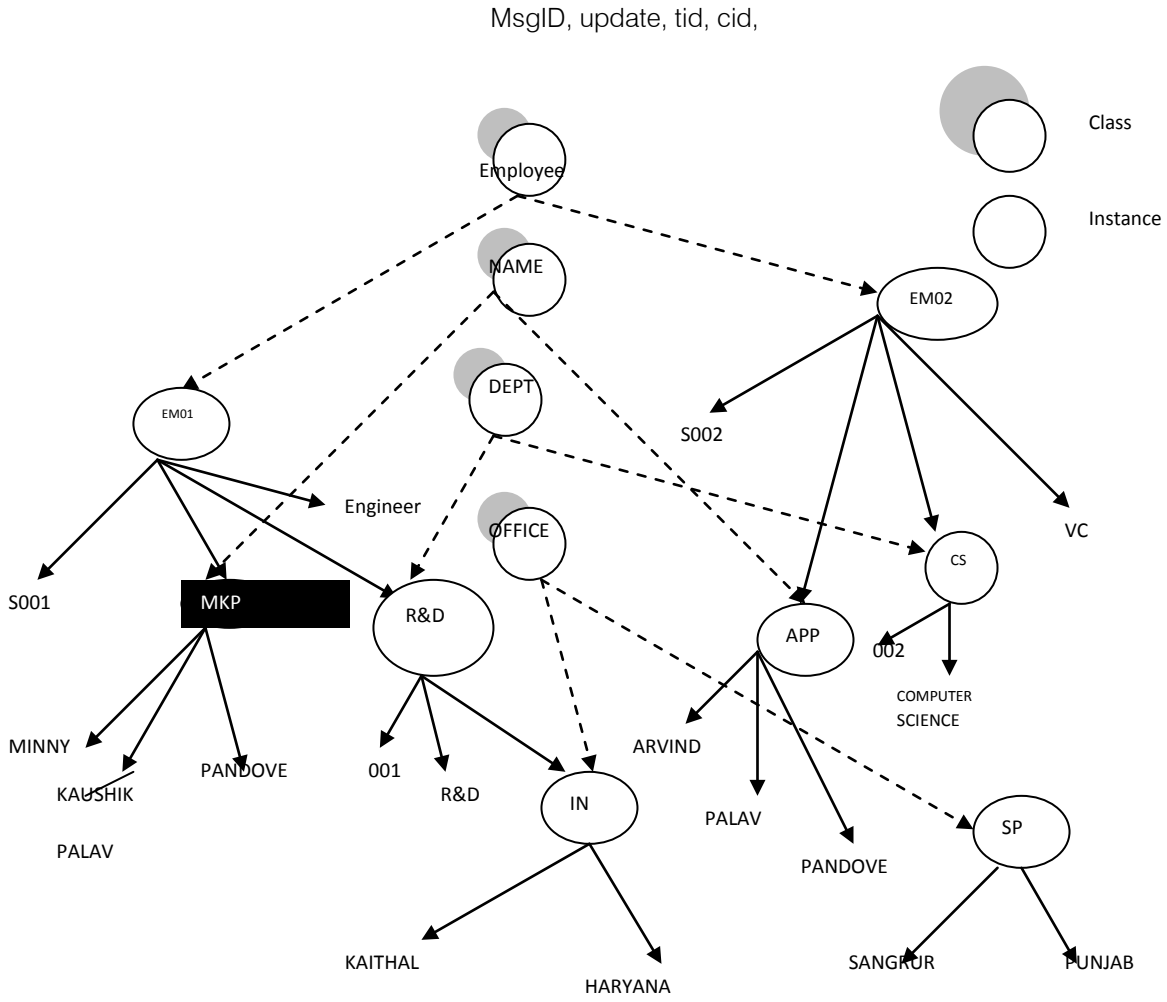


*Figure 4 :* The Graphical Representation of Instance Modification update for Example 4 (From 2 to Figure4)

where ui denotes the i-th attribute name to be updated and vi denotes the new value of ui. For example, assume that the value of attribute name in instance MKP is changed from KAUSHIK to PALAV. The data collector will detect it and send a transaction message ( S001, update, NAME, MKP, {(MKP PALAV)}) to the warehouse. The view -maintenance algorithm for processing the above instance-modification transaction message is proposed as follows.

**The maintenance algorithm for instance modification update:**

**Input :** data warehouse W (C, V, I) and a modified instance tid of class cid.
**Output :** A revised data warehouse W' (C, V, I').

**Step1.** Receive an instance-modification transaction message which is formed from the data collector.
**Step2**. Search the data warehouse W for instances tid; If instance tid exists in W, do the next step; Otherwise, set W' = W and exit the algorithm.
**Step3.** For the instance tid in the warehouse, change its attribute values according to the transaction message.
**Step4.** Check whether the instance tid satisfies the conditions of the views V which refer to the class cid; If the instance satisfies the condition of at least one view, keep instance tid in I of the warehouse W; Otherwise, remove tid from I in the warehouse W. After Step 4, the attribute values of instance tid have been modified in the data warehouse. An example is given below to demonstrate the instance -modification algorithm.

**Example4**, Continuing Example 3, assume that the attribute values of an instance have been modified in the source database, and the transaction message is formed as (S001, update, Name, MKP, {(NAME PALAV)}). This message is processed by the instance -modification algorithm as follows.

**Step1.** Receive the transaction message (S001, update, NAME, MKP, {(NAME PALAV)}) from the data collector.
**Step 2.** Since the instance A1 exists in the warehouse W, the algorithm executes Step 3.
**Step 3**. Change the value of attribute NAME of the instance MKP from KAUSHIK to PALAV.
**Step 4**. Since A1 satisfies the condition of the view MKP Employee, it is kept in W.
The graphical representation of the warehouse after the attribute value of instance A1 has been changed is shown in Figure 4.

## VII. Implementation

Oracle has Object oriented capabilities. This example demonstrates how to

- create a type
- derive a new type from it
- and how to store instance of this type in a table

```
SQL> Create type name as object (
first char(20),
middle char(20),
last char(20)
);
/
SQL> Create type office as object (
State char(30),
city char(30)
);
/
SQL> Create type dept as object (
deptid char(30),
deptname char(40),
deptoffice  office,
counter int
);
/
SQL> Create type employee as object (
 EmpID char(20),
 EmpName name,
 Empdept dept,
 Emptitle char(10),
 counter int
);
SQL> Create table emp of employee;
SQL>insert into emp values (employee ('EM01', name('minny, 'kaushik', 'pandove'),
  dept ('s001', 'R&D','kaithal',' Haryana', 'Engineering'));
SQL> Alter type office add attribute (country char(20)) cascade
SQL> Update type name set middle ='palav' where employeeid='EM01'
```

## VIII. Conclusion

The research of object oriented data warehousing is current topic so, there are many important issues which are yet to be explored. For online processing modification maintenance in object oriented data warehousing is very important. Modification maintenance of the data warehouse is very important to accuracy of the on-line analytical processing. In this paper, we have discussed the concept of object oriented data warehouse and modification maintenance algorithms to maintain the consistency between the data warehousing and the source databases. They are instance insertion, instance modification alters and instance modification update. Although the proposed algorithms can be used to make object oriented data warehousing practical.

## References References Referencias

1. S. Chaudhuri and U. Dyal, "An overview of Data warehousing and OLAP Technology", ACM

SIGMOD record, vol 21, no.1.
2. W.C. Chen, T.P Hong and W.Y. Lin, "Object-Oriented data warehousing and its maintenance technology", technical note, 1998.
3. W.C. Chen, T.P Hong and W.Y. Lin, "View maintenance in an object oriented data warehousing", Proceeding of the Fourth international conference on computer science and informatics, North Carolina, USA, pp 353-356, 1998.
4. N.Hyun, "Efficient self maintenance of materialized views", Technical note, 1996.
5. W.C chen, "Object Oriented data warehousing and its maintenance technologies, Master Dissertation, I-Shou University, Taiwan, R.O.C April, 1999.
6. Dobrovnik M., Eder J.: Logical data independence and modularity through views in OODBMS. Proceedings of the Engineering Systems Design and Analysis Conference, Vol. 2, 1996, pp. 13-20
7. Dobrovnik M., Eder J.: Partial Replication of Object–Oriented Databases. Proceedings of the Second East-European Conference on Advances in Databases and Information Systems – ADBIS'98. Poland, 1998, LNCS No. 1475, pp. 260-271
8. Eder J., Frank H., Liebhart W.: Optimization of Object–Oriented Queries by Inverse Methods. Proceedings of East/West Database Workshop, Austria, 1994
9. Gupta A., Mumick I.S. (eds.): Materialized Views: Techniques, Implementations, and Applications. The MIT Press, 1999.
10. GRAY W. HANSEN.JAMES V.HANSEN, "Data Base Management System".
11. N.Huyn. "Efficient view self maintenance" Proceeding of the ACM Workshop on Materialized Views, Monteral, Canada, June7, 1996.
12. N. Huyn. "Efficient self-maintenance of materialized views," Technical Note, 1996.
13. N. Huyn. "Multiple-view self-maintenance in data warehousing environments," Proceedings of the 23d CZDB Conference, Athens, Greece, 1997.
14. W. H. Inmon and C. Kelley, RdbMS: Developing The Data Warehouse, QED Publishing Group, Boston, Massachusetts, 1 993.
15. W. J. Labio and H. Garcia-Molina. "Efficient snapshot differential algorithms for data warehousing,'' Procwdings of VZDB Conference, Mumbai, India, September 1996, pp.
16. W. J. Labio, Y. Zhuge, J. L. Wiener, H. Gupta, H. Garcia- Molina and J. Widom. "The W " S prototype for data warehouse creation and maintenance," Proceedings of the ACMSIGMOD Conference, Tuscon, Arizona, May 1997.
17. W. Kim, "Modern Database Systems", ACM Press. New York, New York, 1995.
18. I Mumick, D. Quass and B. Mumick. "Maintenance of data cubes and summary tables in a warehouse," Proceedings of the ACM SIGMOD Conference, Tucson, Arizona, May, 1997.
19. P. O'Neil and D. Quass. "Improved query performance with variant indexes," Proceedings of the ACM SIG\IOD Conference, Tucson, Arizona, May, 1997.
20. D. Quass. Maintenance expressions for views with aggregation," Proceedings of the ACM Workshop on Materialized Ems, Montreal, Canada, June 7, 1996.
21. D. Quass, A. Gupta, I. S. Mumick, and J. Widom "Making views self-maintainable for data warehousing," Proceedings of the Conference on Parallel and Distributed Information Systems, Miami Beach, FL, December 1996.
22. Y. G. Ra, E. A. Rundensteiner, "A Transparent Schema- Evolution System Based on Object-Oriented View Technology", IEEE Transaction on Knowledge and Data Engineering, Vol. 9, No. 4,600-624.
23. E. A. Rundensteiner, "A Methodology for supporting Multiple Views in Object-Oriented Databases", Proceedings of 18th International Conference on Very Large Data Bases 1992, Vancouver, Canada, 187-198.
24. M. H. Scholl, C. Laasch, M. Tresch, "Updateable Views in Object-Oriented Databases", Second International Conference on Deductive and Object-Oriented Databases, Munich, Germany, 1991, 189-207.
25. J. Widom, "Research problems in data warehousing, "Proceedings of the 41h Int'l Conference on Information and Knowledge Management, November 1995.
26. Y. Zhuge and H. Garica-Molina. "Graph structured views and their incremental maintenance," Proceedings of the International Conference on Data Engineering, Orlando, FL, 1998.
27. Y. Zhuge, H. Garcia-Molina, J. Hammer and J. Widom, "View maintenance in a warehousing environment" Proceedings of the ACM SIGMOD Conference, San Jose, California, May 1995.

-

This page is intentionally left blank