# Implementation of Object Oriented Data Warehousing using a Narrower Compassed Data Model in Oracle 10g

Dr. Pushpa Suri
Associate Professor
Deptt. of Computer Sc. & Applications
Kurukshetra University
Kurukshetra, India

Meenakshi Sharma
Sr. Lecturer
Deptt of Computer Science & Engineering
Haryana College of Technology and Management
Kaithal , India

## ABSTRACT
A data warehouse (DW) is a database used for reporting Paper describes Object Oriented Data Warehousing using a narrower compassed data model. The data is offloaded from the operational systems for reporting. The data may pass through an operational data store for additional operations before it is used in the Data warehousing for reporting. An Object Oriented Data Warehouse system includes a data warehouse and underlying data sources. A narrower compassed model is used for storing the data in Object Oriented Data warehouse. Oracle 10g Language is specifically used for the programming. Object Oriented Data Warehousing provides complex objects which include multiple atomic types and user defined object types. The steadiness between the data warehouse and the source databases is maintained by certain algorithms such as insertion, deletion and update. In the past research work on data warehouse was primarily focused on relational data models. The concept of object oriented data warehousing is introduced and implementation of Object oriented data warehouse in oracle 10g. Data model will form new classes according to the definition of views such that query performance & security can be improved.

## Keywords
Data warehousing, Oracle 10g, object oriented database, maintenance, narrower compassed data model

## 1. INTRODUCTION
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.

A 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, we propose another model, called the narrower compassed model, which combines all the attributes of the different employees defined in a single employee class. According to the new class the instances are generated and stored in the warehouse. The performance of queries would be better in such narrower compassed data model. Three incremental maintenance algorithms are maintained for the regularity between the data warehouse and the source databases.

## 2. OBJECT ORIENTED DATA WAREHOUSE
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. To meet this objective, it is necessary to extract, transform, and transport data from 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.

## 3. COMPASSED DATA MODEL

The compassed data model [17, 18] combines all the attributes of different classes defined in a view into a single newly generated class. The instances are thus generated and stored in the data warehouse according to the new class. The performance of queries will be better the uncompressed data model since the classes and instances were stored according to the views definition. The compressed data model need not record the relationships of original classes and will thus have a higher security than the uncompressed data model since the referential relationship of the classes is hidden in the data warehouse. However, the procedure for view maintenance using the compressed data model is a little more complex than using the uncompressed data model since the instances are more complex. Since classes and instances are stored in the data warehouse, the problem of incrementally maintaining these objects is thus critical to the usage of the data.

## 4. 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 c is a quadruple {cid, ca, ct, cm} where cid ε ID, ca = < $ca_{1,}$ …… $ca_n$> with $ca_i \in$ A and i= 1 to n, ct = < $ct_{1,}$ …… $ct_n$ > with $ct_j$ ε T and j=1 to n, and cm $\subseteq$ M.

**Example 1:** In this example four classes, EmpInfo , Name, Dept, and Classes. The class EmpInfo has three attributes, EmpID, EmpInfo, EmpClass and one method Counter(). In this example attribute EmpID is a character type, the attribute EmpName and EmpClass are of the type Name and Classes, which are classes.In the class EmpInfo in this example, cid = EmpInfo, ca= {EmpID, EmpName, EmpClass}, ct= {char(10), Name, Classes}, and cm={Counter())}. Let C be the set of classes defined in the source database. An instance is created by referring to a class and inheriting characteristics from the class.

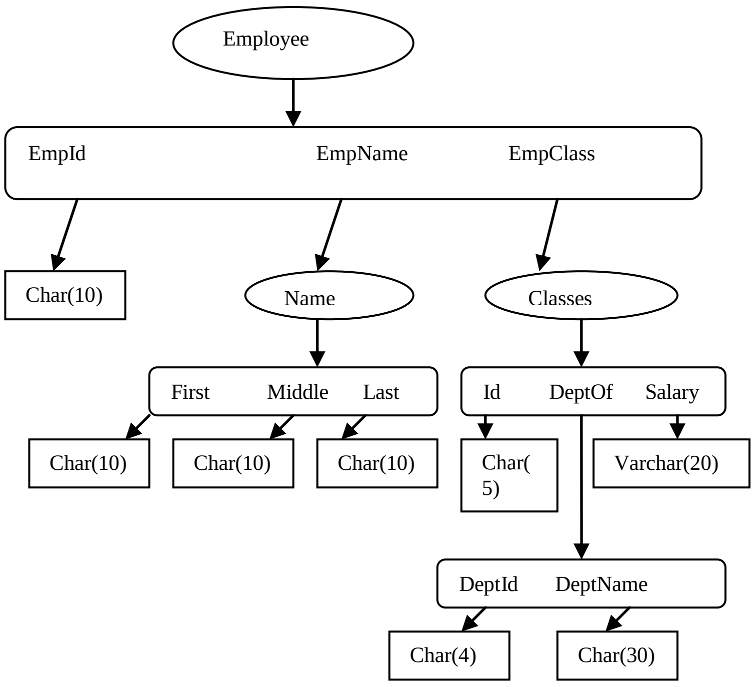

**Figure 1: A Graphical image of classes**

B. Definition (Instance)

An instance t ={tid, ta, tv,tm,tc}is created and inherits from a certain class cid={cid, ca, ct, cm} such that tid $\in$ 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 1, assume that two instances are created by and referring to class Emp*Dept.* One is called *RS* with attribute values (001, Research Scholar) the other is called *IT with* attribute values (002, Information Technology). Similarly, assume that two instances, *A1* and *B1* respectively, with attribute values (001, CS, 20,000) and (102, IT, 15000) are created by referring to the class *Classes.* Assume that two instances, *WCC* and *TPH* respectively, with attribute values (Meenakshi, Arvind, Pandove) and (Palav, Arvind, Pandove) are created by referring to the class *Name.* Also assume that two instances, *STOl* and *STO2* respectively, with attribute values (57001, WCC, Al) and (53001, TPH, Bl) are created by referring to the class *EmpInfo.*

## 5. 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 department as object (
1.        deptid char(4),
2.        deptname char(30)
3.        );
4.        /
SQL> Create type classes as object (
1.        id char(5),
2.        deptof department,
3.        salary varchar(20)
```

```
4.          );
5.          /
SQL> Create type name as object (
1.          first char(10),
2.          middle char(10),
3.          last char(10)
4.          );
5.          /
SQL> Create type employee as object (
1.          EmpID char(10),
2.          EmpName name,
3.          EmpClass classes
4.          );
SQL> Create table employee1 of employee;
SQL>insert into employee1 values (
1.     employee (‘s001’,
2.    name(‘meenakshi, ‘arvind’, ‘pandove’),
3.    classes (‘c001’, department (‘d001’, ‘research’),
     ’20      lac’));
```

## 6. 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 Iid 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 Iid 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’)

## 7. INSTANCE DELETION

We have a source database now we want to delete an existing instance Iid from the source database. For this transaction Msg is formed and for view maintance this Msg is sent from the data collectors to the Data warehouse. Instance is not immediately deleted from the source database. Instance is removed by the data collectors only after the maintenance in the data warehouse has been performed. The proposed syntax for the instance deletion is MID, delete , Iid, Cid.

**The algorithm of maintenance for instance deletion**

**Step1:** A source database, receive an instance deletion transaction Message, which is received from the data collector.

**Step2:** Make the view definition to find the definition which is refers to the class Cid. By applying select, from and where operations two new views are found TF by using FROM operation and Ts by using SELECT, WHERE operation.

**Step3:** In TF for each view V, delete instance with Iid in the latter part of its identifier.

**Step4:** In TS , for each view V denote all the attributes appearing in SELECT and WHERE operation and denote them $V_B$ .

**Step5:** Send a request to the data collector for the contents of B from the instance tid or from its descending instances.

**Step6**: Receive the contents of B from the data collector.

**Step7:** For each view v in TS. Remove the instances that satisfy the conditions of v using the contents of B.

**Step8**: After the instance deletion maintenance procedure, all necessary instances are removed from the data warehouse to maintain consistency and the instance tid has been actually deleted from the source database.

## 8. CONCLUSION

In this paper we have discussed the concept of object oriented data warehousing .this is being done with the help of narrower compassed data model to speed up the query performance in data warehouse. The regularity between the data warehouse and the data base of the system has been improved by the three algorithms as insertion, deletion and modification. Data warehouses contain consolidated data from many databases and other external data sources, spanning long time periods, and augmented with summary information. There are many new challenges in designing, creating and maintaining large data warehouses. There are several new promising research issues some of which are related to problems that the database community has worked on for several years, but others are only just beginning to be addressed. The benefit with this model is that the user can easily access the data needed from the data warehouse and the model can also keep the original inheritance structure by copying it from the source database. In the future, we will try to enhance our research for more friendly environmental domains so as to reach the more flexible object oriented data warehousing system.

## 9. REFERENCES

[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. [28] Y. Zhuge, H. Garcia-