ISSN:2231-0711
www.ijcset.net

Oracle Support For User-Defined Objects Across Distributed Object-Relational Database

Clarence J M Tauro[1], Dr.N Ganesan[2], Jayakumar J[3], SaurabhRelan[4]

[1]clarence.tauro@res.christuniversity.in*
Centre for Research, Christ University
Hosur Road, Bangalore, India

[2]ganeshnj@gmail.com
Director (MCA),RICM, Bangalore, India

[3]jayakumr.jr@gmail.com
[4]er.saurabh2005@gmail.com
Department of Computer Science, Christ University
Hosur Road, Bangalore, India

***Abstract*—Thispaper gives an overview of accessing the user-defined objects across the Oracle ORDB in a distributed environment, which allows application to access the data & objects from a network of two or more homogenous ORDB that natively supports object-model.**

Keywords- ORDB; Distributed database; Object-Model; User-defined Objects;

I. INTRODUCTION

In Modern Information System (MIS) the databases are kept in distributed environment for storing data and to serve large number of information access requested by online users. In a homogenous distributed database system, collection of same databases are stored in single or many machine in a network with, each database having distinct data and supporting database objects or each having a replication of master database objects in all the database[4]. ORDB can be defined as Object Oriented system over the relational database system with collection of object-elements (classes and objects), operations performed on the object elements and traditional relational data [1].

When ORDB is implemented in distributed environment, thelocal databases will copy the database objects from master database to maintain the integrity of the distributeddatabase and to provide valid data when accessed [2]. While copying it will copy the all the user-defined objects along with the others database objects.

Rest of the paper is organized as follows, in Section IIwe discuss about the user-defined types in Oracle database. Section III explains the implementation of ORDB in distributed environment. Section IVdescribes about the accessing the user defined objects in distributed database. Finally, Section V gives conclusion.

II. USER-DEFINED OBJECTS IN ORACLE

Oracle 11g is an object-relational database management system (ORDBMS) [4, 8], which means that users can define additional kinds of dataspecifying both the structure of the data and the ways of operating on it and use these types within the relational model. This approach adds value to the data stored in a database. User-defined data types make it easier for application developers to work with complex data such as images, audio, and video. Object types store structured business data in its natural form and allow applications to retrieve it that way. For that reason, applications developed using Object-Oriented Programming techniques works efficiently.

III. USER-DEFINED DATA TYPES

Object types and other user-defined datatypes allowusers to define their own data type that model the structure and behavior of the data in their applications. There are two categories of user-defined datatypes [8]; Object types and Collection types.

A. Object types

Object types are abstractions of the real-world entities for example, purchase orders that application programs deal with. An object type is a schema object with three kinds of components; A*Name*, which serves to identify the object type uniquely within that schema Attributes, which model the structure and state of the real-world entity. *Attributes*, they are built-in types or other user-defined types.*Methods*, they are functions or procedures written in PL/SQL and stored in database, or written in a language such as C or Java and stored externally. Methods implement operations the application can perform on the real-world entity.An object type is a template, defining it does not result in storage allocation [4].

A structured data unit that matches the template is called an object.Figure1 gives an example of how to define object types called `EXTERNAL_PERSON`, `LINEITEM`, and `PURCHASE_ORDER`.The object types `EXTERNAL_PERSON`and `LINEITEM`have attributes of built-in types. The object type `PURCHASE_ORDER`has a complex

*Senior Courseware Developer at SpringSource (a division of vmware)

structure, which closely matches the structure of real purchase orders.The attributes of PURCHASE_ORDERare ID, CONTACT, and LINEITEMS. The attribute CONTACT is an object, and the attribute LINEITEMS is a nested table.

The commands in Figure 1 are used to create the above object types [3].

```
CREATE TYPE external_personAS OBJECT (
nameVARCHAR2(30),
phone       VARCHAR2(20) );

CREATE TYPE lineitem AS OBJECT (
item_nameVARCHAR2(30),
quantity    NUMBER,
unit_priceNUMBER(12,2) );


CREATE TYPE lineitem_tableAS TABLE OF lineitem;

CREATE TYPE purchase_order AS OBJECT (
id          NUMBER,
contactexternal_person,
lineitemslineitem_table,
  MEMBER FUNCTIONget_valueRETURN NUMBER );
```

Figure 1: Example of Object Type

In this example given in Figure 1,which does not show how to specify the body of the method GET_VALUE, nor does it show the full complexity of a real purchase order.

The CONTACT table is a relational table which keeps track of CONTANCTS with an object type defining one of its columns. Figure 2 shows the command to create the CONTACT relational table [3];

```
CREATE TABLE contacts (
contactexternal_person
dateDATE );
```

Figure 2: Example of Object Table

Types of Methods

Methods of an object type model the behavior of objects. The methods of an object type broadly fall into these categories:

- A *Member method* is a function or a procedure that always has an implicit SELF parameter as its first parameter, whose type is the containing object type.
- A *Static method* is a function or a procedure that does not have an implicit SELF parameter. Such methods can be invoked by qualifying the method with the type name, as in TYPE_NAME.METHOD(). Static methods are useful for specifying user-defined constructors or cast methods.
- *Comparison methods* are used for comparing instances of objects.

Oracle supports the choice of implementing type methods in PL/SQL, JAVA, and C.In the example given in Figure 1, PURCHASE_ORDER has a method named GET_VALUE. Each purchase order object has its own GET_VALUE method. For example, if x and y are PL/SQL variables that hold purchase order objects and w and z are variables that hold numbers, the following two statements given in Figure 3, can leave w and z with different values:

```
w = x.get_value();
z = y.get value();
```

Figure 3: Commands to swap values

After those statements, w has the value of the purchase order referred to by variable x; z has the value of the purchase order referred to by variable y.

The term x.GET_VALUE() is an invocation of the method GET_VALUE. Method definitions can include parameters, but GET_VALUE does not need them, because it finds all of its arguments among the attributes of the object to which its invocation is tied. That is, in the first statement given in Figure 3, it computes its value using the attributes of purchase order x. In the second it computes its value using the attributes of purchase order y. This is called the selfish style of method invocation.

Every object type also has one implicitly defined method that is not tied to specific objects, the object type's constructor method.Every object type has a system-defined constructor method; that is, a method that makes a new object according to the object type's specification. The name of the constructor method is the name of the object type. Its parameters have the names and types of the object type's attributes. The constructor method is a function. It returns the new object as its value.

The expression in Figure 4 represents a purchase order object with the following attributes;

```
purchase_order (1000376,
external_person ("John Smith","1-800-555-1212"),
NULL )
```

Figure 4: Object of Purchase Order

Values of, ID is '1000376', EXTERNAL_PERSONis ("John Smith", "1-800-555-1212"), LINEITEMSis Null

The expression EXTERNAL_PERSON ("John Smith", "1-800-555-1212") is an invocation of the constructor function for the object type EXTERNAL_PERSON. The object that it returns becomes the contact attribute of the purchase order.

B. Collection Types

Each collection type describes a data unit made up of an indefinite number of elements, all of the same datatype. The collection types are array types and table types.Array types and table types are schema objects. The corresponding data units are called VARRAYs and nested tables. When there is no danger of confusion the collection types are often referred as VARRAYs and nested tables.

Collection types have constructor methods. The name of the constructor method is the name of the type, and its argument is a comma separated list of the new collection's

elements. The constructor method is a function. It returns the new collection as its value.An expression consisting of the type name followed by empty parentheses represents a call to the constructor method to create an empty collection of that type. An empty collection is different from a null collection[4].

IV. STORING USER-DEFINED OBJECTS IN DATABASE

In Oracle ORDBMS the user-defined objects are stored in special kind of Object Table and provide a relations view of the attributes of those objects [4].If the object appears in the column of the table then it is called Column Objects, if it appears in the whole table then it is called Row Objects.In Object Table each row object & column object will be associated with a unique system-generated 16 byte logical object identifier (OID). Oracle uses this OID references to fetch and navigate the objects when user requested.

V. IMPLEMENTATION OF ORDB IN DISTRIBUTED ENVIRONMNET

In a homogenous replicated distributed database system, Oracle databases are kept in network with each having the same copy of database objects [4]. Figure 5 illustrates the distributed database environment with three databases having distinct global database names`A, B, C` connected to each other.

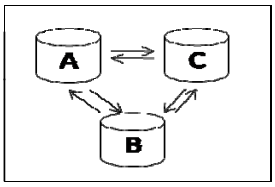

Figure5: Distributed Environment with three Sites

The databases A, B, and C s having the replication of same database objects to provide the data locally to user for faster access and provide data even if local system fails but other system are accessible. Any change made in any one of the database will be copied to other two databases to maintain the integrity of the database.

A. Client/Server Architecture

The distributed database system will be designed using traditional Client/Server Architecture, in which system will become Client when replicating the database object from another database, which is acting as Sever [4, 7]. If `A` is replicating the data from `B`, then `A` becomes the Client and `B` becomes the Server.

B. Database Links

Database links are the pointers that define a one-way communication from an Oracle database to another database [4, 8]. The link defined in `A` for `B` will be stored in data dictionary table and is only accessible by the users of `A`. To make a two-way communication the database link should be created in both the database.

The SQL in Figure 6 creates the database link`DBlink_A_to_B` from `A` to `B`

```
Create public database link DBlink_A_to_B using B
```

Figure 6: Creating a Database Link from `A` to `B`

C. Replication Architecture

In Replicated Distributed Database the Oracle allows replication of following database objects [4];

- Tables
- Indexes
- Views and Object Views
- Packages and Package Bodies
- Procedures and Functions
- User-Defined Types and Type Bodies
- Triggers
- Synonyms
- Indextypes
- User-Defined operators

In distributed database replication can happen in single master replication or multi master replication [4, 5]. In *Single Master Replication*,all the local sitesconnected in distributed environment will have the copy of the single master with each local site having database link to master site. In *Multi Master Replication*, all the sites are master site with each having data that as to be updated in all the other sites. In Multimaster replication each site will have database links to all individual master sites totaling in `N-1` database links, where `N` is the number of master sites connected in distributed environment.

VI. REPLICATION OF USER DEFINED OBJECTS IN DISTRIBUTED DATABASE

As discussed in Section 4 the user-defined objects are stored in Oracle tables. As the basic requirement for the replication, the database object structure must exist in each site and each site must have exact structure. Like wise to replicate schema objects based on user-defined types, the user-defined types themselves must exist, and must be exactly same, at the replication site.

A. Conditions for replicating user-defined objects [4]

a. The entire replication site must have the same OID, owner and type name for a replicated user-defined type.
b. For object type, all the replication sites must agree on the order and data type of the attributes in the object type.
c. All the replication sites must agree on hash code of the user-defined type. Hash code will be assigned by Oracle database after examining the type attributes, order of attributes, and type name.

B. Condition for replicating object tables [4]

a. The OID of an object table must be the same at all the replication sites.
b. The OID of each row object in an abject table must be same at all the replication sites.

VII. CONCULSION

In this paper we have given overview of how Oracle ORDB gives flexibility in modeling the real world entity in database level using user-defined types, which can be accessed by users directly without help of any external language.

When Oracle database is implemented in distributed multimaster replication environment it supports replication of database objects as well as replication of user objects in distributed systems. This support from Oracle ORDB gives new way in,storing and accessing the real world objects which are used in MIS.

ACKNOWLEDGMENT

We are heartily thankful to Prof. Jibrael and Prof. Joy Paulose, Department of Computer Science, Christ University, whose encouragement, guidance and support enabled us to develop an understanding of the subject.

REFERENCES

[1] Justus S., “Metrics for Object Relational Databases”, International Conference on Recent Trends in Information Systems, National Engineering College, TN, IND, Jan 2006.

[2] Trajanov, D.; Glamocanin, V.; Dusko, S.; Stojkovska, B.; Petar, I.; , "Distribution and replication of object oriented data for FRIENDS ," Power System Technology, 2000. Proceedings. PowerCon 2000. International Conference on , vol.2, no., pp.655-659 vol.2, 2000

[3] Pardede, E.; Rahayu, J.W.; Taniar, D.; , "New SQL standard for object-relational database applications," Standardization and Innovation in Information Technology, 2003. The 3rd Conference on , vol., no., pp. 191- 203, 22-24 Oct. 2003

[4] Oracle Document Library.http://www.oracle.com/pls/db111/homepage

[5] Tong Cui; , "LDAP Directory Template Model on Multi-master Replication," Services Computing Conference (APSCC), 2010 IEEE Asia-Pacific , vol., no., pp.479-484, 6-10 Dec. 2010

[6] Filip, I.; Vasar, C.; Robu, R.; , "Considerations about an Oracle database multi-master replication," Applied Computational Intelligence and Informatics, 2009. SACI '09. 5th International Symposium on , vol., no., pp.147-152, 28-29 May 2009

[7] Zubi, Z.S.; , "On distributed database security aspects," Multimedia Computing and Systems, 2009. ICMCS '09. International Conference on Multimedia Computing and Systems, vol., no., pp.231-235, 2-4 April 2009

[8] Kevin Loney, “Oracle Database 11g:The Complete Reference”, Tata McGraw Hill Education Private Limited 2010, ISBN: 0070140790