Practical - 3

Aim: Implementation of ORDBMS using ADT(Abstract Data Types), References, Inheritance etc.

ORDBMS [Object-Relational Database Management System]

An **object relational database management system** (ORDBMS) is a database management system that is similar to a relational database, except that it has an object-oriented database model. This system supports objects, classes and inheritance in database schemas and query language.

- PostgreSQL.
- Informix by IBM.
- SQL Server by Microsoft.
- Greenplum Database by Pivotal Software.

Advantages of ORDBMSs

There are following advantages of ORDBMSs:

Reuse and Sharing: The main advantages of extending the Relational data model come from reuse and sharing. Reuse comes from the ability to extend the DBMS server to perform standard functionality centrally, rather than have it coded in each application.

Increased Productivity: ORDBMS provides increased productivity both for the developer and for the, end user

Use of experience in developing RDBMS: Another obvious advantage is that .the extended relational approach preserves the significant body of knowledge and experience that has gone into developing relational applications. This is a significant advantage, as many organizations would find it prohibitively expensive to change. If the new functionality is designed appropriately, this approach should allow organizations to take advantage of the new extensions in an evolutionary way without losing the benefits of current database features and functions.

Disadvantages of ORDBMSs

The ORDBMS approach has the obvious disadvantages of complexity and associated increased costs. Further, there are the proponents of the relational approach that believe the· essential simplicity' and purity of the relational model are lost with these types of extension.

ORDBMS vendors are attempting to portray object models as extensions to the relational model with some additional complexities. This potentially misses the point of object orientation,

highlighting the large semantic gap between these two technologies. Object applications are simply not as data-centric as relational-based ones.

Abstract Data Types (ADT):- By using abstract data types, which are user-defined types, together with various routines, you can uniquely define and use data with complex structures and perform operations on such data. When you define a column as having an abstract data type, you can conceptualize and model its data based on object-oriented concepts. In addition, by applying object oriented software development techniques, you can reduce the workload for database design, UAP development, and maintenance

REF Function:- In Oracle PL/SQL, REF data types are pointers that uniquely identify a piece of data as an object. A reference can be established between an existent valid object and a table or type attribute using the REF pointer data type. An attribute referring to a nonexistent object leads to "dangling" situation. Note that a NULL object reference is different from a Dangling Reference. To insert data into a ref column, the REF function is used to get an object instance reference.

DEREF Function:- DEREF returns the object reference of argument expr, where expr must return a REF to an object. If you do not use this function in a query, then Oracle Database returns the object ID of the REF instead.

Inheritance:- Inheritance is based on a family tree of object types that forms a type hierarchy. The type hierarchy consists of a parent object type, called a supertype, and one or more levels of child object types, called subtypes, which are derived from the parent.

A] Abstract Data Type :

Creating type : type type_name_finny_48 and type_address_finny_48

```
create type type_name_finny_48 as object
  (
  fname varchar2(20),
  mname varchar2(20),
  lname varchar2(20)
  );
  /
```

```
SQL> create type type_name_finny_48 as object
  2    (
  3    fname varchar2(20),
  4    mname varchar2(20),
  5    lname varchar2(20)
  6    );
  7  /

Type created.
```

create type type_address_finny_48 as object
(
street varchar2(20),
city varchar2(20),
pincode number(10)
);
/

```
SQL> create type type_address_finny_48 as object
  2    (
  3    street varchar2(20),
  4    city varchar2(20),
  5    pincode number(10)
  6    );
  7  /

Type created.
```

Creating table customer_finny:

create table customer_finny_48(
c_id number(5) primary key,
c_name type_name_finny_48,
c_add type_address_finny_48,
c_phone_number number(20));

```
SQL> create table customer_finny_48(
  2    c_id number(5) primary key,
  3    c_name type_name_finny_48,
  4    c_add type_address_finny_48,
  5    c_phone_number number(20));

Table created.
```

Inserting the records in the customer_finny table:-
Code :-
SQL> Insert into customer_finny_48 values(1, type_name_finny_48('Finny','George','Sabu'), type_address_finny_48('Rajaji','Dombivili',479763),9084662124);
Output :-

```
SQL> Insert into customer_finny_48 values(1, type_name_finny_48('Finny','George','Sabu'),type_address_finny_48('Rajaji',
'Dombivili',479763),9084662124);

1 row created.
```

Displaying all records:
Code :-
Select * from customer_finny_48;
Output :-

```
SQL> Select * from customer_finny_48;

      C_ID
----------
C_NAME(FNAME, MNAME, LNAME)
--------------------------------------------------------------------------------------------------------
------------------------------
C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------------------------------
------------------------------
C_PHONE_NUMBER
--------------
         1
TYPE_NAME_FINNY_48('Finny', 'George', 'Sabu')
TYPE_ADDRESS_FINNY_48('Rajaji', 'Dombivili', 479763)
    9084662124
```

Displaying only street of the customer of c_id=1 :
Code :-
select c.c_add.street from customer_finny_48 c where c.c_id = 1;
Output :-

```
SQL> select c.c_add.street from customer_finny_48 c where c.c_id = 1;

C_ADD.STREET
--------------------
Rajaji
```

Viewing in depth structure of the table :
Code :-
set describe depth 2;
desc customer_finny;
Output :-

```
SQL> set describe depth 2;
SQL> desc customer_finny_48;
 Name                                                                                    Null?    Type
 ----------------------------------------------------------------------------------------- -------- ----------------------------
-----------------------------
 C_ID                                                                                    NOT NULL NUMBER(5)
 C_NAME                                                                                           TYPE_NAME_FINNY_48
   FNAME                                                                                          VARCHAR2(20)
   MNAME                                                                                          VARCHAR2(20)
   LNAME                                                                                          VARCHAR2(20)
 C_ADD                                                                                            TYPE_ADDRESS_FINNY_48
   STREET                                                                                         VARCHAR2(20)
   CITY                                                                                           VARCHAR2(20)
   PINCODE                                                                                        NUMBER(10)
 C_PHONE_NUMBER                                                                                   NUMBER(20)
```

Displaying first name, middle name, last name of the customer1_sachin table:
Code :-
SQL> select c.c_name.fname || ‘ ‘||c.c_name_mname|| ‘ ‘ ||c.c_name.lname from customer_finny_48 c;
Output :-

```
SQL> select c.c_name.fname || ' '|| c.c_name.mname || ' ' || c.c_name.lname from customer_finny_48 c;

C.C_NAME.FNAME||''||C.C_NAME.MNAME||''||C.C_NAME.LNAME
------------------------------------------------------------
Finny George Sabu
```

B] REF :

Code :-

```
SQL> create or replace type Animal_TY as object(
  2  breed varchar2(25),
  3  name varchar2(25),
  4  birthdate date);
  5  /

Type created.
```

```
SQL> create table Animal_finny of Animal_TY;

Table created.
```

```
SQL> insert into Animal_finny values(Animal_TY('Monkey','Franky','01-APR-02'));

1 row created.

SQL> insert into Animal_finny values(Animal_TY('Cat','Timmy','04-DEC-00'));

1 row created.

SQL> insert into Animal_finny values(Animal_TY('Dog','Tom','15-JAN-05'));

1 row created.
```

```
SQL> select REF(A) from Animal_finny A;

REF(A)
--------------------------------------------------------------------------------------------------------
0000280209D1A80D5A8EBF4412A1D7F9C932339365C77DD0CB26CC45B6ACCB2748B60EF0710041C3B10000
00002802090DFF5112DEBF44CEBE5641D7254AF239C77DD0CB26CC45B6ACCB2748B60EF0710041C3B10001
000028020984EE1854C9274F6DA1BC413FFAE7BF06C77DD0CB26CC45B6ACCB2748B60EF0710041C3B10002
```

C] DEREF :

Code :-

```
SQL> create table keeper_finny(
  2  keepername varchar2(25),
  3  animalkept REF Animal_TY
  4  );

Table created.
```

```
SQL> describe keeper_finny;
 Name                                                  Null?    Type
 ----------------------------------------------------- -------- ------------------------------
 KEEPERNAME                                                     VARCHAR2(25)
 ANIMALKEPT                                                     REF OF ANIMAL_TY
```

```
SQL> insert into keeper_finny select 'Finny', REF(A) from Animal_Finny A where name = 'Tom';

1 row created.

SQL> select * from keeper_finny
  2  ;

KEEPERNAME
-------------------------
ANIMALKEPT
--------------------------------------------------------------------------------------------------------
Finny
000022020884EE1854C9274F6DA1BC413FFAE7BF06C77DD0CB26CC45B6ACCB2748B60EF071
```

```
SQL> select keepername, DEREF(K.animalkept) from keeper_finny K;

KEEPERNAME
-------------------------
DEREF(K.ANIMALKEPT)(BREED, NAME, BIRTHDATE)
--------------------------------------------------------------------------------
Finny
ANIMAL_TY('Dog', 'Tom', '15-JAN-05')


SQL>
```


D] INHERITANCE :-

Code :-

```
CREATE Or Replace TYPE AddressType_finny AS OBJECT (
   street VARCHAR2(15),
    city   VARCHAR2(15),
  state  CHAR(2),
   zip    VARCHAR2(5)
  )
```

Output :-

```
SQL> CREATE Or Replace TYPE AddressType_finny AS OBJECT (
  2      street VARCHAR2(15),
  3       city   VARCHAR2(15),
  4     state  CHAR(2),
  5      zip    VARCHAR2(5)
  6     )
  7  /

Type created.
```

Code :-

```
CREATE Or Replace TYPE PersonType_finny_48 AS OBJECT (
   id        NUMBER,
    first_name VARCHAR2(10),
   last_name  VARCHAR2(10),
  dob       DATE,
    phone      VARCHAR2(12),
   address    AddressType_finny
  ) NOT FINAL;
/
```

Output :-

```
SQL> CREATE Or Replace TYPE PersonType_finny_48 AS OBJECT (
  2  id NUMBER,
  3  first_name VARCHAR2(10),
  4  last_name  VARCHAR2(10),
  5  dob DATE,
  6  phone VARCHAR2(12),
  7  address AddressType_finny
  8  ) NOT FINAL;
  9  /

Type created.
```

Code :-

```
CREATE Or replace TYPE business_PersonType_finny_48 UNDER PersonType_finny_48 (
     title   VARCHAR2(20),
   company VARCHAR2(20));
```

Output :-

```
SQL> CREATE Or replace TYPE business_PersonType_finny_48 UNDER PersonType_finny_48 (
  2        title   VARCHAR2(20),
  3     company VARCHAR2(20));
  4  /

Type created.
```

Code :-

```
CREATE TABLE object_business_customers_finny_48 OF business_PersonType_finny
_48;
```

Output :-

```
SQL> CREATE TABLE object_business_customers_finny_48 OF business_PersonType_finny_48;

Table created.
```

Code ;-

```
 INSERT INTO object_business_customers_finny_48 VALUES (
   business_PersonType_finny_48(1, 'Finny', 'Sabu', '07-FEB-2000', '9086523718',
  AddressType_finny('rajaji', 'Dombivili', 'MA', '12345'),'System Eng', 'TATA')
   );
```

Output :-

```
SQL> INSERT INTO object_business_customers_finny_48 VALUES (
  2     business_PersonType_finny_48(1, 'Finny', 'Sabu', '07-FEB-2000', '9086523718',
  3    AddressType_finny('rajaji', 'Dombivili', 'MA', '12345'),'System Eng', 'TATA')
  4     );

1 row created.
```

Code :-

```
select * from object_business_customers_finny_48;
```

Output :-

```
SQL> select * from object_business_customers_finny_48;

        ID FIRST_NAME LAST_NAME  DOB       PHONE
---------- ---------- ---------- --------- ------------
ADDRESS(STREET, CITY, STATE, ZIP)
--------------------------------------------------------------------------------------------------------
------------------------------
TITLE                COMPANY
-------------------- --------------------
         1 Finny      Sabu       07-FEB-00 9086523718
ADDRESSTYPE_FINNY('rajaji', 'Dombivili', 'MA', '12345')
System Eng           TATA
```

Conclusion :- Successfully implemented ADT(Abstract Data Types),References and Inheritance.