Practical - 3

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

Theory:

ORDBMS:

ORDBMS stands for Object-Relational Database Management System. It provides all the facilities of RDBMS with the additional support of object-oriented concepts. The concept of classes, objects, and inheritance are supported in this database. It is present at the ground level between the RDBMS and OODBMS. This data can be manipulated by using any query language. It is complex because it has to take care of both Relational database concepts as well as Object Oriented concepts.

Advantages of ORDBMS:-

- **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 is 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 ORDBMS:-

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

1. **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.

2. **Inheritance**

Inheritance enables you to share attributes between objects such that a subclass inherits attributes from its parent class. OracleAS TopLink provides several methods to preserve inheritance relationships and enables you to override mappings that are specified in a superclass or to map attributes that are not mapped in the superclass. Subclasses must include the same database field (or fields) as the parent class for their primary key (although the primary key can have different names in these two tables). As a result, when you are mapping relationships to a subclass stored in a separate table, the subclass table must include the parent table primary key, even if the subclass primary key differs from the parent primary key.

3. **REF & DREF Functions:-**

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 a "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.

```
Enter user-name: user15b@sem1
Enter password:

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
```

IMPLEMENTATION OF ADT

Creating type: type_name_siddhartha and type_address_siddhartha

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

Type created.

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

Type created.
```

Creating table customer1_siddhartha

```
SQL> create table customer1_siddhartha
  2  ( cid number(5) PRIMARY KEY,
  3  cname type_name_siddhartha,
  4  caddress type_address_siddhartha,
  5  ccontact number(20));

Table created.
```

Describing the structure of table customer1_siddhartha

```
SQL> desc customer1_siddhartha;
 Name
ull?    Type
 -------------------------------------------
------- ------------------------------------
 CID
OT NULL NUMBER(5)
 CNAME
        TYPE_NAME_SIDDHARTHA
 CADDRESS
        TYPE_ADDRESS_SIDDHARTHA
 CCONTACT
        NUMBER(20)
```

Inserting value into main table

SQL> insert into customer1_siddhartha values(101,type_name_siddhartha('Adam','Kumar','Shukla'),type_address_siddhartha('chembur','mumbai',400071),1234567890);

Displaying record of table customer1_siddhartha;

```
SQL> select * from customer1_siddhartha;

       CID
----------
CNAME(FNAME, MNAME, LNAME)
--------------------------------------------------------------------------------
CADDRESS(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  CCONTACT
----------
       101
TYPE_NAME_SIDDHARTHA('Adam', 'Kumar', 'Shukla')
TYPE_ADDRESS_SIDDHARTHA('chembur', 'mumbai', 400071)
1234567890
```

Displaying only street of the customer with id=101

```
SQL> select c.caddress.street from customer1_siddhartha c where cid=101;

CADDRESS.STREET
--------------------
chembur
```

Viewing in depth structure of table customer1_siddhartha.

```
SQL> set describe depth 2;
SQL> desc customer1_siddhartha;
 Name
ull?    Type
 ------------------------------------------------------------
------- ------------------------------------------------------
 CID
OT NULL NUMBER(5)
 CNAME
        TYPE_NAME_SIDDHARTHA
   FNAME
        VARCHAR2(20)
   MNAME
        VARCHAR2(20)
   LNAME
        VARCHAR2(20)
 CADDRESS
        TYPE_ADDRESS_SIDDHARTHA
   STREET
        VARCHAR2(20)
   CITY
        VARCHAR2(20)
   PINCODE
        NUMBER(10)
 CCONTACT
        NUMBER(20)
```

Displaying fname, mname and lname of customer1_siddahrtha table.

```
SQL> select n.cname.fname||' '||n.cname.mname||' '||n.cname.lname from customer1_siddhartha n;

N.CNAME.FNAME||''||N.CNAME.MNAME||''||N.CNAME.LNAME
--------------------------------------------------------------
Adam Kumar Shukla
```

<u>IMPLEMENTATION OF REF AND DREF FUNCTION:</u>

REF FUNCTION

Creating course object

```
SQL> create or replace type course as object
  2  (
  3  coursename varchar2(10),
  4  duration number(2),
  5  coursefee number(3)
  6  );
  7  /

Type created.
```

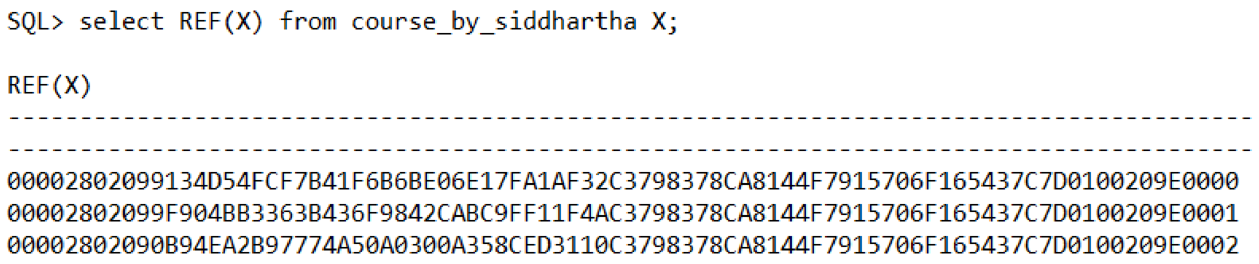
Creating table course_by_siddhartha of object course.

```
SQL> create table course_by_siddhartha of course;

Table created.

SQL> insert into course_by_siddhartha values(course('zerotohero',3,999));

1 row created.
```

Inserting values into table course_by_siddhartha using object course.

```
SQL> insert into course_by_siddhartha values(course('growskill',2,799));

1 row created.

SQL> insert into course_by_siddhartha values(course('betteryou',1,499));

1 row created.
```

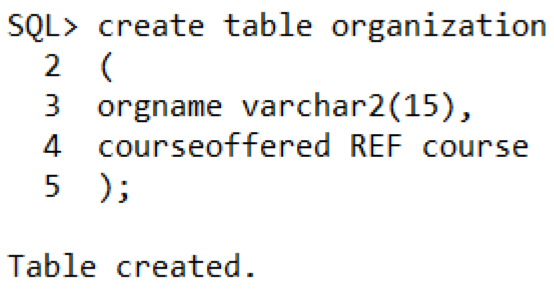
Displaying memory location using REF FUNCTION of table course_by_siddahrtha;



```
SQL> select REF(X) from course_by_siddhartha X;

REF(X)
------------------------------------------------------------------------------
------------------------------------------------------------------------------
00002802099134D54FCF7B41F6B6BE06E17FA1AF32C3798378CA8144F7915706F165437C7D0100209E0000
00002802099F904BB3363B436F9842CABC9FF11F4AC3798378CA8144F7915706F165437C7D0100209E0001
00002802090B94EA2B97774A50A0300A358CED3110C3798378CA8144F7915706F165437C7D0100209E0002
```

DEF FUNCTION

Creating table organization

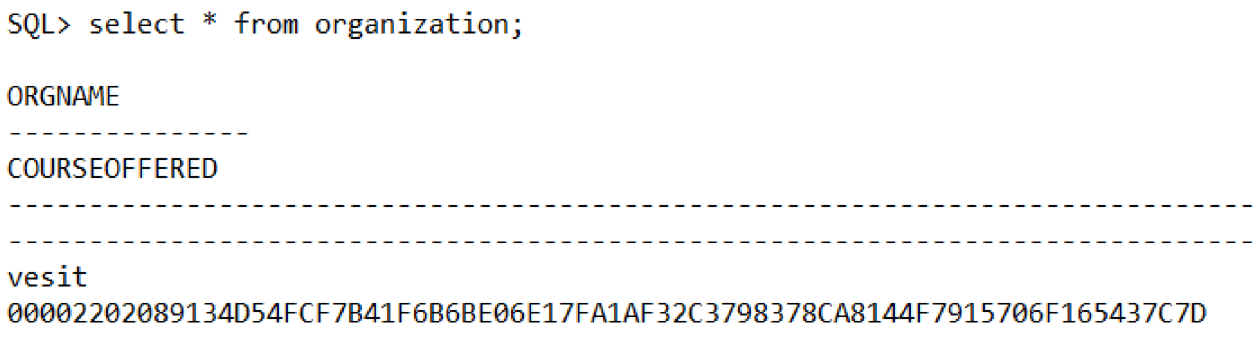


```
SQL> create table organization
  2  (
  3  orgname varchar2(15),
  4  courseoffered REF course
  5  );

Table created.
```

Inserting values into table organization

SQL> insert into organization select 'vesit',REF(x) from course_by_siddhartha X where duration=3;

Displaying table organization



```
SQL> select * from organization;

ORGNAME
---------------
COURSEOFFERED
------------------------------------------------------------------------------
------------------------------------------------------------------------------
vesit
00002202089134D54FCF7B41F6B6BE06E17FA1AF32C3798378CA8144F7915706F165437C7D
```

Displaying records using DREF function

```
SQL> select orgname,DEREF(d.courseoffered)from organization d;

ORGNAME
---------------
DEREF(D.COURSEOFFERED)(COURSENAME, DURATION, COURSEFEE)
----------------------------------------------------------------
----------------------------------------------------------------
vesit
COURSE('zerotohero', 3, 999)
```

IMPLEMENTATION OF INHERITANCE

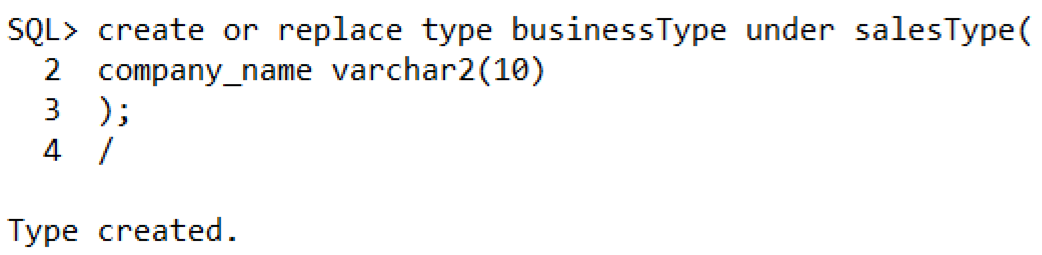
Creating object product type.

```
SQL> create or replace type productType as object(
  2  product_id number(3),
  3  product_name varchar2(10),
  4  product_category varchar2(10));
  5  /

Type created.
```

Creating object sales type which is extending product type.

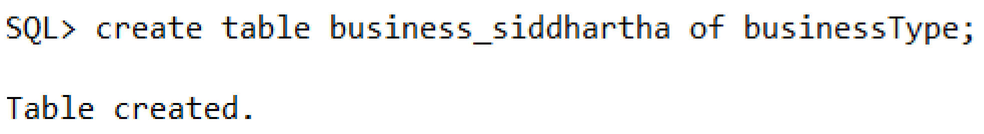
```
SQL> create or replace type salesType as object(
  2  sales_id number(3),
  3  sales_name varchar2(10),
  4  sales_contact number(10),
  5  sales_product productType)not final;
  6  /

Type created.
```

Creating object business type which is inheriting from sales type



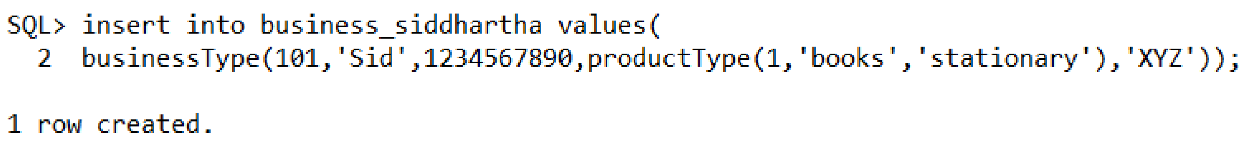
```
SQL> create or replace type businessType under salesType(
  2  company_name varchar2(10)
  3  );
  4  /

Type created.
```

Creating table business_siddhartha of object business Type.



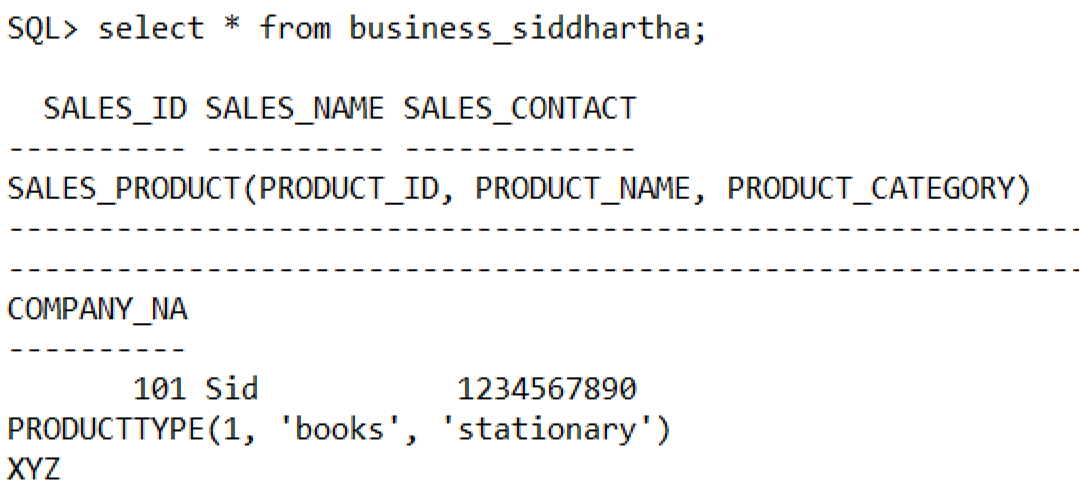
```
SQL> create table business_siddhartha of businessType;

Table created.
```

Inserting values into table business_siddhartha.



```
SQL> insert into business_siddhartha values(
  2  businessType(101,'Sid',1234567890,productType(1,'books','stationary'),'XYZ'));

1 row created.
```

Displaying records of table business_siddhartha.



```
SQL> select * from business_siddhartha;

  SALES_ID SALES_NAME SALES_CONTACT
---------- ---------- -------------
SALES_PRODUCT(PRODUCT_ID, PRODUCT_NAME, PRODUCT_CATEGORY)
------------------------------------------------------------
------------------------------------------------------------
COMPANY_NA
----------
       101 Sid           1234567890
PRODUCTTYPE(1, 'books', 'stationary')
XYZ
```

Conclusion: I have successfully implemented ORDBMS concepts including ADT, REF-DREF and Inheritance.