

# Object-Relational Databases 42901

## Database Design And Implementation Assignment

### (Online Music Database)

#### Student Details

Name	Student_ID
Piyush Kumar	13677493
Vraj Mehta	13488642

Oracle Username – 13488642 (Assignment Implemented)

## Introduction

This assignment explains the design for Online music database with the help of logical model and relational model in Barker notations with fully normalized form in section 2.1. We will also be creating an object relational design implementing collections including variable array and nested table (features of object-relational database). In order to deal with disk type, we will be using super-type feature for “Mp3” and “Audio CDs & Vinyl Albums” as these both entities will be the sub type of Albums entity. After that we will insert all the data as provided for the database in section 2.2.

Apart from that, we will go through “Methods and Queries” in section 2.3, in which queries will show how to use max, avg, intersect, union, nvl and many other pre-defined functions. With the help of queries, we will represent implementation of object types and fetching the values from the types using treat function. These queries will also demonstrate how to implement method and view. Not only implementation, it will also describe how to modify view.

In the Final section “2.4 Discussion”, we will compare our both designs (Relational and Object-Relational design) from section 2.1 and 2.2 and will highlight the advantages and features of both design and which is better to be implemented in the given scenario for online music database (OMDB).

**Table of Contents:**

Object-Relational Databases 42901	1
<b>DATABASE DESIGN AND IMPLEMENTATION ASSIGNMENT 1</b>	
(Online Music Database)	1
Introduction	1
2.1 Entity-Relationship Model and Relational Design	3
(a). Part I – Logical Model	3
(b). Part II – Relational Model	4
2.3 OMDB Methods and Queries	7
2.4 Discussion	19
Appendix	20
2.2 OMDB Object-Relational Design and Implementation	20

## 2.1 Entity-Relationship Model And Relational Design

### (A). Part I – Logical Model

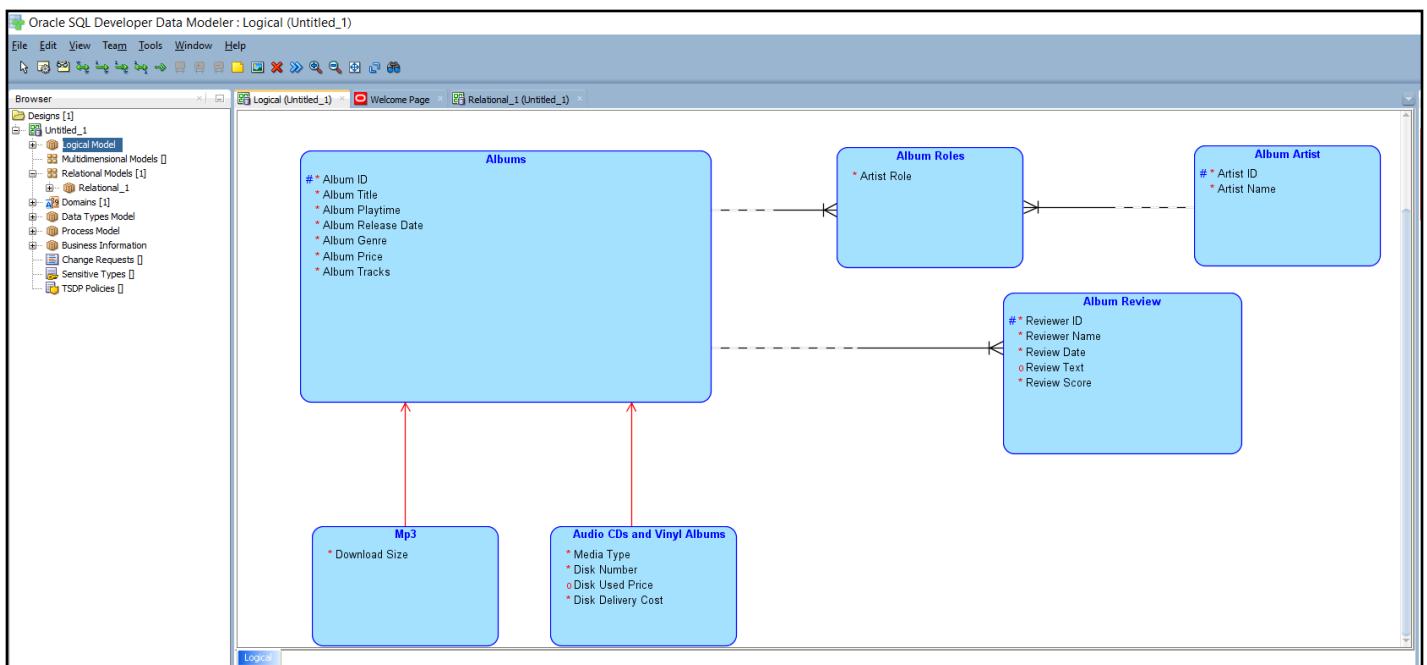
ER (Entity Relationship) diagram is kind of a flowchart that represents how different entities within a system are interrelated. This logical model is considered as the easiest way to understand the relationship between the objects such as tables.

According to the given scenario of online music database (OMDB), Albums is the main table containing Album Title, Album Playtime, Album Release Date, Album Genre, Album Price, Album Tracks columns. There are two subtypes of the Albums table and that are “Mp3” and another one is “Audio CDs and Vinyl Albums” because according to the given dataset it is clear that a single album can have different media types and that is the reason these media types are the subtype of Albums table. Mp3 subtype contains one column and that is Download Size. Whereas, in case of Audio CDs and Vinyl disk subtype there are Media Type, Disk Number, Disk Used Price, Disk Delivery Cost. Due to different column exists in “Mp3” and “Audio CDs and Vinyl Albums” both are considered as different subtypes of Albums. There is another table exist and that is Album Artists that has Artist Name, Artist ID.

The **relationship** between Albums and Album Roles is one to many because there can be many roles for one album. The relationship between Album Artist and Album Roles is one to many as there can multiple roles for a single Artist. Album Reviews is one table that consists of Reviewer Name, Review Date, Review Text, Review Score. The relationship between Albums and Album Reviews is one to many.

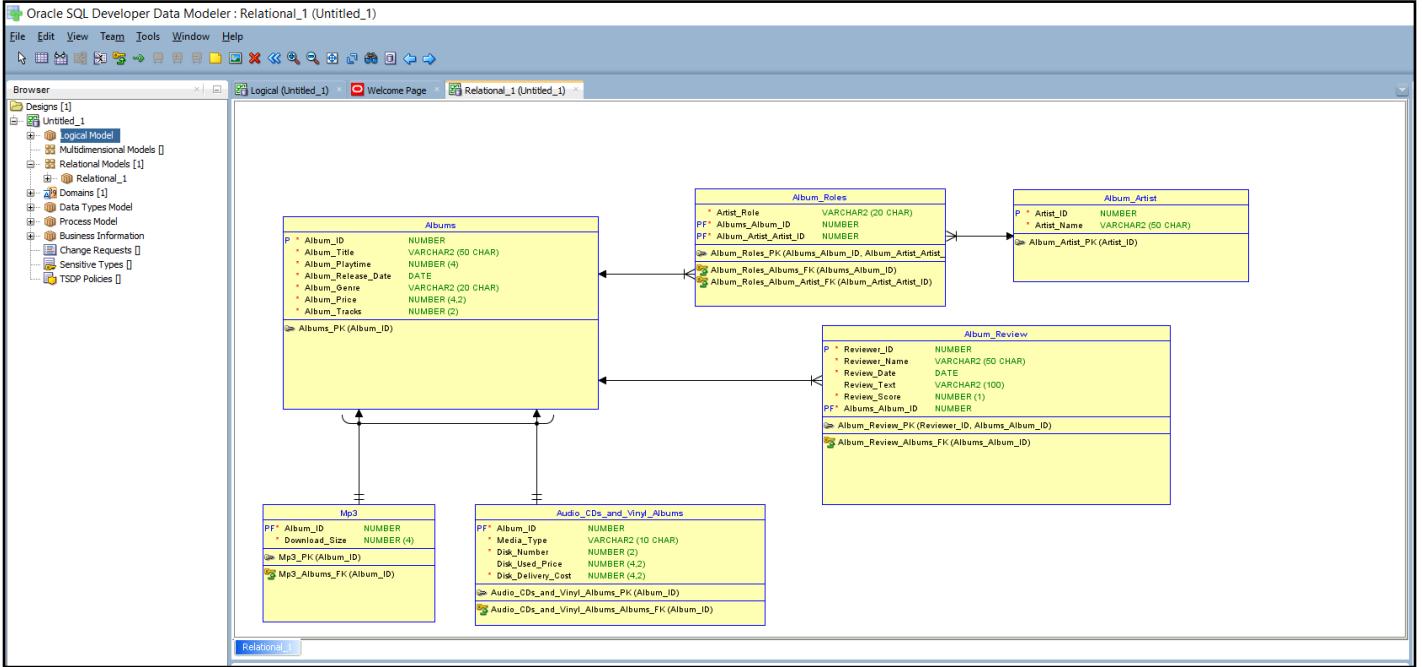
Album ID is the **unique identifier** for Albums. Similarly, Artist ID & Reviewer ID are the unique identifiers for Album Artists and Album Review respectively. We have created unique identifier as no suitable attribute exists.

Below if the Figure for the **Logical Model** for the given scenario. It also shows whether a attribute should be mandatory or not.



## (B). Part II – Relational Model

With the help of “Logical designed Entity Relationship Model” we have engineer it to “Relational Model”.



As Above Table Diagram or Relational Model is not completely normalized. In order to Normalize this Model, we need to find out the functional dependencies and then remove them from this model.

So, Table is “**Albums**” that contains (Albums\_ID, Album\_Title, Album\_Playtime, Album\_Release\_Date, Album\_Genre, Album\_Price, Album\_Tracks) attributes has dependencies such as

**Album\_Title**  $\rightarrow$  Album\_Playtime, Album\_Tracks, Album\_Genre

**Albums\_ID, Album\_Title**  $\rightarrow$  Album\_Release\_Date, Album\_Price

Normalizing the given Scenario.

**Step 1.** Find merged minimal cover of Functional Dependencies, which contains:

**Album\_Title**  $\rightarrow$  Album\_Playtime, Album\_Tracks, Album\_Genre

**Albums\_ID, Album\_Title**  $\rightarrow$  Album\_Release\_Date, Album\_Price

Initially relation [1] contains the original table, with the Functional Dependencies above

**Check:** Checking whether table relation [1] is in BCNF

The Functional Dependency [Album\_Title  $\rightarrow$  Album\_Playtime, Album\_Tracks,

**Album\_Genre**] violates BCNF as the Left-Hand Side is not super key. Table is split into the two below:

**relation [2] = (Album\_Title, Album\_Playtime, Album\_Tracks, Album\_Genre )**

With Functional Dependencies: There are no functional Dependency in relation 2.

**relation [3] = (Albums\_ID, Alum\_Title, Album\_Release\_Date, Album\_Price)**

With Functional Dependencies: There are no functional Dependency in relation 3.

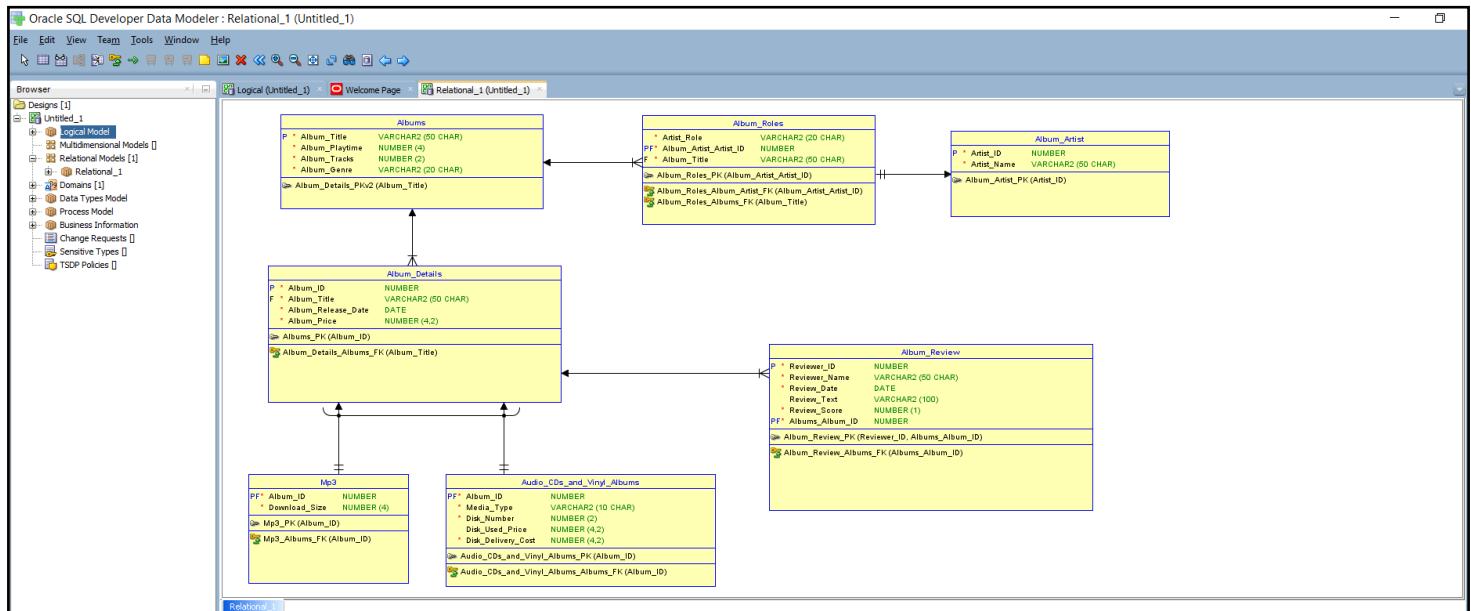
**Check:** Checking whether table relation [2] is in normalized form

Yes, Relation [2] – (Album\_Title, Album\_Playtime, Album\_Tracks, Album\_Genre) is in fully normalized form because it follows 1NF (as it has all atomic values), 2NF (as there is no partial Dependency), 3NF (As there is no Transitive Dependency), BCNF (as in this case now LHS is a super key).

**Check:** Checking whether table relation [3] is in normalized form

Yes, Relation [3] – (Albums\_ID, Alum\_Title, Album\_Release\_Date, Album\_Price) is in fully normalized form because it follows 1NF (as it has all atomic values), 2NF (as there is no partial Dependency), 3NF (As there is no Transitive Dependency), BCNF (as in this case now LHS is a super key).

## Applying Normalization to Relational Model –



As table Albums is now split into two new entities that are Albums that contains (Album\_Title, Album\_Playtime, Album\_Tracks, Album\_Genre) and Album\_Details that contains (Album\_ID, Album\_Title, Album\_Release\_Date, Album\_Price). Due to the splitting of table Album, there will be certain changes in relationships between entities.

- Albums entity has one to many relation with Album\_Roles.
- Albums table has one to many relation with Album\_Details.
- Albums\_Details entity has one to many relation with Album\_Review.
- MP3 entity has one to one relation with Album\_Details.
- Audio\_CDs and Vinyl\_Albums table has one to one relation with Album\_Details.

## 2.3 Omdb Methods And Queries

- Give album title, album release date and album price of all Neil Young's albums released after 1st January 2015.

**Query:**

```
select distinct a.albumTitle, a.albumReleaseDate, a.albumPrice
from albums a,
TABLE (a.albumArtists) r
where r.artistName = 'Neil Young' and a.albumReleaseDate>'1-Jan-2015'
```

**Result:**

The screenshot shows the Oracle SQL Developer interface with the following components visible:

- Connections** pane: Shows connections named ELL and uts. The uts connection is selected.
- Worksheet**: Contains the SQL query:
 

```
select distinct a.albumTitle, a.albumReleaseDate, a.albumPrice
from albums a,
TABLE (a.albumArtists) r
where r.artistName = 'Neil Young' and a.albumReleaseDate>'1-Jan-2015'
```
- Script Output** and **Query Result** panes: Both show the output of the query. The **Query Result** pane displays the results in a table format:
 

ALBUMTITLE	ALBUMRELEASEDATE	ALBUMPRICE
1 Best of Neil Young	21/FEB/19	17.5

 The status message "All Rows Fetched: 1 in 0.105 seconds" is shown above the results.
- Messages - Log** pane at the bottom.
- Messages**, **Logging Page**, and **Statements** tabs at the bottom of the messages pane.

2. Give album title and artist name for albums released only in MP3 format. Order by album title.

**Query:**

```
select distinct a.albumTitle, r.artistName
from albums a,
TABLE (a.albumArtists) r
where value(a) is of (mp3_type)
order by a.albumTitle
```

**Result:**

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections sidebar shows an Oracle connection named 'uts'. The central area has two tabs: 'Worksheet' and 'Query Builder'. The 'Worksheet' tab contains the SQL query:

```
select distinct a.albumTitle, r.artistName
from albums a,
TABLE (a.albumArtists) r
where value(a) is of (mp3_type)
order by a.albumTitle
```

Below the query is a 'Query Result' window titled 'Query Result x'. It displays the results of the executed query:

ALBUMTITLE	ARTISTNAME
1 B.B. King Greatest Hits	B.B. King
2 Best of Neil Young	Neil Young
3 Bob Dylan's Greatest Hits	Bob Dylan
4 Harvest (2009 Remaster)	Neil Young
5 Other Peoples Lives	Stats
6 Sketches of Spain	Miles Davis
7 The Essential Bob Dylan	Bob Dylan

The 'Worksheet' tab also shows the message 'All Rows Fetched: 7 in 0.064 seconds'.

**3. Give lowest rated MP3 album (i.e. album with the lowest average review score). Show album title and the average score. Exclude albums with only one review.**

**Query:**

```
select a.albumTitle, avg(r.reviewScore)
from albums a,
TABLE(a.albumReviews) r
where value(a) is of (mp3_type)
group by a.albumTitle
having avg(r.reviewScore) = (select min(avg(r.reviewScore))
                             from albums a,
                             TABLE(a.albumReviews) r
                             where value(a) is of (mp3_type)
                             group by a.albumTitle
                             having count(r.reviewScore)>1)

intersect

select a.albumTitle, avg(r.reviewScore)
from albums a,
TABLE(a.albumReviews) r
where value(a) is of (mp3_type)
group by a.albumTitle
having count(r.reviewScore)>1
```

The screenshot shows the Oracle SQL Developer interface. The left sidebar contains connections to 'ELL' and 'uts', reports, and SSH hosts. The main area has tabs for 'Worksheet' and 'Query Builder'. The 'Worksheet' tab displays the complex SQL query. The 'Query Result' tab shows the output:

ALBUMTITLE	AVG(R.REVIEWSCORE)
1 The Essential Bob Dylan	5

Below the tabs, it says 'All Rows Fetched: 1 in 0.106 seconds'. The bottom navigation bar includes 'Messages', 'Logging Page', and 'Statements'.

**Result:**

**4. Are there any albums released on all media, i.e. on MP3, audio CD and vinyl? Show album title and order by album title.**

**Query:**

```
select a.albumTitle
from albums a
where value(a) is of (mp3_type)
intersect
select a.albumTitle
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Vinyl'
intersect
select a.albumTitle
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Audio CD'
order by albumTitle
```

**Result:**

The screenshot shows the Oracle SQL Developer interface. The left sidebar contains the 'Connections' tree, which is expanded to show 'Tables', 'Views', 'Indexes', 'Procedures', 'Functions', 'Operators', 'Queues', 'Triggers', 'Types', and 'Sequences'. Below it are sections for 'Reports' (containing 'All Reports', 'Analytic View Reports', 'Data Dictionary Reports', 'Data Modeler Reports', 'OLAP Reports', 'TimesTen Reports', and 'User Defined Reports') and 'SSH Hosts' (containing 'SSH Hosts'). At the bottom is the 'REST Development' section with 'REST Data Services'. The main area is divided into three tabs: 'Worksheet' (containing the query code), 'Script Output' (showing the execution status 'All Rows Fetched: 4 in 0.009 seconds'), and 'Query Result' (displaying a table with one column 'ALBUMTITLE' containing four rows: '1 Bob Dylans Greatest Hits', '2 Harvest (2009 Remaster)', '3 Sketches of Spain', and '4 The Essential Bob Dylan'). The bottom navigation bar includes tabs for 'Messages', 'Logging Page', and 'Statements'.

ALBUMTITLE
1 Bob Dylans Greatest Hits
2 Harvest (2009 Remaster)
3 Sketches of Spain
4 The Essential Bob Dylan

**5. Implement the method discountPrice() that returns a discounted price using the following business rule:**

- a. for audio CDs released more than one year ago the discount is 20%
- b. for vinyl records released more than one year ago the discount is 15%
- c. for MP3 downloads released more than two years ago the discount is 10%

**Method:**

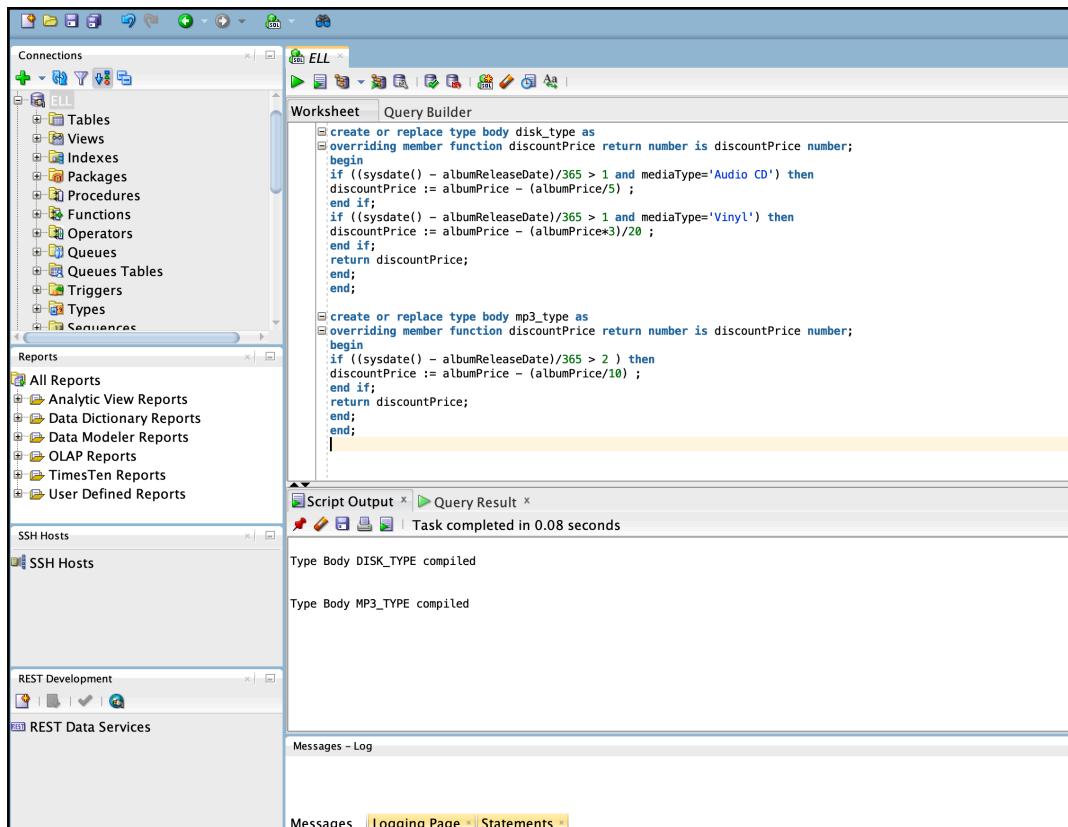
```

create or replace type body disk_type as
overriding member function discountPrice return number is discountPrice
number;
begin
if ((sysdate() - albumReleaseDate)/365 > 1 and mediaType='Audio CD') then
discountPrice := albumPrice - (albumPrice/5) ;
end if;
if ((sysdate() - albumReleaseDate)/365 > 1 and mediaType='Vinyl') then
discountPrice := albumPrice - (albumPrice*3)/20 ;
end if;
return discountPrice;
end;
end;

-----
create or replace type body mp3_type as
overriding member function discountPrice return number is discountPrice
number;
begin
if ((sysdate() - albumReleaseDate)/365 > 2 ) then
discountPrice := albumPrice - (albumPrice/10) ;
end if;
return discountPrice;
end;
end;
```

**Note:** This method returns **NULL** if discount is not applied according to the given conditions

**Result:**



EMENTATION

- 6. Create a view all\_albums that includes the columns: album title, media type ('MP3', 'Vinyl', 'Audio CD'), album price, and discount (album price – discount price). Use this view to find the album that received the largest discount; show all view columns.**

**View:**

```
create view all_albums(albumTitle,diskType,albumPrice,discount)
as
select a.albumTitle,'MP3', a.albumPrice, nvl(a.albumPrice -
a.discountPrice(),0)
from albums a
where value(a) is of (mp3_type)
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
nvl(a.albumPrice - a.discountPrice(),0)
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Vinyl'
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
nvl(a.albumPrice - a.discountPrice(),0)
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Audio CD'
```

**Note: NVL function** to check If discount() function return NULL and change it to 0.

### Result:

The screenshot shows the Oracle SQL Developer interface. On the left, there's a tree view of database objects under the connection 'ELL'. The 'Tables' node is expanded. In the center, the 'Worksheet' tab is active, displaying the SQL code for creating a view:

```

create view all_albums(albumTitle,diskType,albumPrice,discount)
as
select a.albumTitle,'MP3', a.albumPrice, nvl(a.albumPrice - a.discountPrice(),0)
from albums a
where value(a) is of (mp3_type)
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice, nvl(a.albumPrice - a.discountPrice(),0)
from albums a
where value(a) is of (disk_type) and treat(value(a) as disk_type).mediaType='Vinyl'
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice, nvl(a.albumPrice - a.discountPrice(),0)
from albums a
where value(a) is of (disk_type) and treat(value(a) as disk_type).mediaType='Audio CD'

```

Below the code, the 'Script Output' window shows the message: "View ALL\_ALBUMS created." The 'Query Result' window shows the execution time: "Task completed in 0.083 seconds". At the bottom, there are tabs for 'Messages', 'Logging Page', and 'Statements'.

### Query (Album received highest discount):

```

select a.albumTitle,a.diskType,a.albumprice,a.discount
from all_albums a
group by a.albumTitle, a.diskType, a.albumprice,a.discount
having max(a.albumprice - a.discount) = ( select max(a.albumprice -
a.discount)
                                         from all_albums a)

```

**Result:**

The screenshot shows the Oracle SQL Developer interface. On the left, there are several toolbars and panes: 'Connections' (with 'ELL' selected), 'Reports' (listing various report types), 'SSH Hosts' (empty), and 'REST Development' (empty). The main workspace is titled 'ELL' and contains a 'Worksheet' tab with the following SQL query:

```
select a.albumTitle,a.diskType,a.albumprice,a.discount
from all_albums a
group by a.albumTitle, a.diskType, a.albumprice,a.discount
having max(a.albumprice - a.discount) = ( select max(a.albumprice - a.discount)
from all_albums a)
```

Below the worksheet is a 'Script Output' tab showing the execution results:

All Rows Fetched: 1 in 0.049 seconds

ALBUMTITLE	DISKTYPE	ALBUMPRICE	DISCOUNT
1 The Essential Bob Dylan Vinyl		37	5.55

At the bottom, there are tabs for 'Messages', 'Logging Page', and 'Statements'.

7. Now, modify the view all\_albums to also include the column album used price for disks; set album used price to zero for MP3 albums. Use this view to find the most expensive used album; show all view columns.

**View:**

```
drop view all_albums;
```

```

create view
all_albums(albumTitle,diskType,albumPrice,discount,diskUsedPrice)
as
select a.albumTitle,'MP3', a.albumPrice, a.discountPrice(),0
from albums a
where value(a) is of (mp3_type)
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
a.discountPrice(), treat(value(a) as disk_type).diskUsedPrice
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Vinyl'
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
a.discountPrice(), treat(value(a) as disk_type).diskUsedPrice
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Audio CD';

```

## Result:

The screenshot shows the Oracle SQL Developer interface. On the left, the Connections sidebar shows an Oracle connection named 'uts'. The main workspace has a 'Worksheet' tab open with the SQL code for creating the 'all\_albums' view. Below the worksheet is a 'Script Output' window showing the execution results: 'View ALL\_ALBUMS dropped.' followed by 'View ALL\_ALBUMS created.'. At the bottom, there are tabs for 'Messages', 'Logging Page', and 'Statements'.

```

drop view all_albums;

create view all_albums(albumTitle,diskType,albumPrice,discount,diskUsedPrice)
as
select a.albumTitle,'MP3', a.albumPrice, a.discountPrice(),0
from albums a
where value(a) is of (mp3_type)
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
a.discountPrice(), treat(value(a) as disk_type).diskUsedPrice
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Vinyl'
union
select a.albumTitle, treat(value(a) as disk_type).mediaType, a.albumPrice,
a.discountPrice(), treat(value(a) as disk_type).diskUsedPrice
from albums a
where value(a) is of (disk_type) and treat(value(a) as
disk_type).mediaType='Audio CD';

View ALL_ALBUMS dropped.

View ALL_ALBUMS created.

```

**Query (Most expensive used Album):**

```
select a.albumTitle,a.diskType,a.albumprice,a.discount,a.diskUsedPrice
from all_albums a
group by a.albumTitle, a.diskType, a.albumprice,a.discount,a.diskUsedPrice
having max(diskUsedPrice) = ( select max(a.diskUsedPrice)
                                from all_albums a )
```

**Result:**

The screenshot shows the Oracle SQL Developer interface. On the left, there's a sidebar with 'Connections' (selected), 'Tables' (under ELL), 'Views', 'Indexes', 'Procedures', 'Functions', 'Operators', 'Queues', 'Queues Tables', 'Triggers', 'Types', and 'Sequences'. Below that are 'Reports' (All Reports, Analytic View Reports, Data Dictionary Reports, Data Modeler Reports, OLAP Reports, TimesTen Reports, User Defined Reports), 'SSH Hosts' (SSH Hosts), and 'REST Development' (REST Data Services). The main area has tabs for 'Worksheet' (containing the query code) and 'Query Builder'. Below these is a 'Script Output' tab showing the results. The results are presented in a table:

ALBUMTITLE	DISKTYPE	ALBUMPRICE	DISCOUNT	DISKUSEDPRICE
1 Kind Of Blue (Legacy Edition) Vinyl		19.99	2.9985	16.99
2 Kind Of Blue (Legacy Edition) Audio CD		19.99	3.998	16.99

8. Implement the method `containsText (pString1, pString2)` that returns 1 if `pString1` contains `pString2`, and 0 if it does not. Use this method to find albums with reviews that contain the word 'Great'. Show album title, review text, review score. Note that the signature of the `containsText` method is included in the original OMDB script.

**Method:**

```

create or replace type body album_type as
member function containsText(pString1 varchar2, pString2 varchar2) return
integer is bool integer;
begin
if ( INSTR(pString2,pString1) > 0) then
bool :=1;
else
bool :=0;
end if;
return bool;
end;

member function discountPrice return number is discount number;
begin
return null;
end;
end;

```

## Result:

The screenshot shows the Oracle SQL Developer interface. The left sidebar displays connections to 'Oracle Connections' (specifically 'ELL' and 'uts') and various reports. The main workspace is titled 'uts' and contains a 'Worksheet' tab with the following PL/SQL code:

```

create or replace type body album_type as
member function containsText(pString1 varchar2, pString2 varchar2) return
integer is bool integer;
begin
if ( INSTR(pString2,pString1) > 0) then
bool :=1;
else
bool :=0;
end if;
return bool;
end;

member function discountPrice return number is discount number;
begin
return null;
end;
end;

```

Below the worksheet, the 'Script Output' tab shows the message: "Type Body ALBUM\_TYPE compiled". The status bar at the bottom indicates "Task completed in 0.329 seconds".

**Query (Review text that contains 'Great'):**

```
select a.albumTitle, r.reviewText, r.reviewScore
from albums a,
TABLE(a.albumReviews) r
where a.containsText('Great',r.reviewText)=1;
```

**Result:**

The screenshot shows the Oracle SQL Developer interface. The left sidebar contains connections to 'ELL' and 'uts', reports, and SSH hosts. The main area has a 'Worksheet' tab with the executed SQL query. Below it is a 'Script Output' tab showing the execution time and a 'Query Result' tab displaying the fetched data.

ALBUMTITLE	REVIEWTEXT	REVIEWSCORE
1 The Essential Bob Dylan Great compilation of some of his most known songs		5
2 Best of Neil Young Great artist and great music.		5
3 Harvest (2009 Remaster) Great artist and great music.		5

## 2.4 Discussion

There are both advantages and disadvantages of Relational design when compared to Object Relational design.

In object relational databases there are pre-defined join paths by the virtue of class design. Relational is more free-form, one can join anything they want; which is very helpful to the one who is mining data. This freedom to join anything yield better results. In our case, if one wants to add more information regarding Albums, eg Album Tracks or Album Location. One cannot do that in object relational as it is pre-defined (it can be done with relatively expensive operation, the object needs to be deleted and then re-inserted), but it is much more easier in relational design to update such as create extra columns or tables and join them.

Interpretation of object relational database is extremely difficult. For eg to access Album Reviews, one needs to create an object and access it. It is trivial only when you are working with limited dataset. If there are a large database (suppose 100 objects to deal with) with a lot of attributes, it will be difficult to access each of them. Relational databases are easier to access and do manipulation. One can easily create a new table, join and perform operations.

Whereas, there are cases when object relational database represent the real world scenario much better than relational database. For this Online Music Database(OMDB), we were able to restrict the number of artists to maximum of five using nested table. We also used VARRAY's to store Album Reviews. These things are very essential in real world databases, which cannot be implemented in relational databases.

## Appendix

### 2.2 Omdb Object-Relational Design And Implementation

->The script used for creating the OMDB database (omdb.sql).

```
-- create OMDB --
-----
-- drop tables --
drop table albums
/
drop type disk_type
/
drop type mp3_type
/
drop type album_type
/
drop type artist_array_type
/
drop type artist_type
/
drop type review_table_type
/
drop type review_type
/
-- create types --
create or replace type artist_type as object
(artistName      varchar(50),
 artistRole      varchar(25))
/
create type artist_array_type
as varray(5) of artist_type
/
```

```

create or replace type review_type as object
(reviewerName    varchar(25),
 reviewDate      date,
 reviewText      varchar(250),
 reviewScore     number)
/
create or replace type review_table_type as table of review_type
/
create or replace type album_type as object
(albumTitle        varchar(50),
 albumPlaytime    number(3), -- minutes
 albumReleaseDate date,
 albumGenre        varchar(15),
 albumPrice       number(9,2),
 albumTracks      number(2),
 albumArtists     artist_array_type,
 albumReviews     review_table_type,
member function discountPrice return number,
member function containsText (pString1 varchar2, pString2 varchar2) return
integer)
not instantiable not final
/
create or replace type disk_type under album_type
( mediaType        varchar(10),
diskNum          number(2), -- number of disks
diskUsedPrice    number(9,2),
diskDeliveryCost number(9,2),
overriding member function discountPrice return number)
/
create or replace type mp3_type under album_type
(downloadSize   number, -- size in MB
 overriding member function discountPrice return number)
/
-- create tables --
create table albums of album_type
object id system generated
nested table albumReviews store as store_reviews
/

```

### \*Result of the script(create database) execution:

The screenshot shows the Oracle SQL Developer interface with the following details:

- Connections:** Oracle Connections - ELL, uts.
- Reports:** All Reports, Analytic View Reports, Data Dictionary Reports, Data Modeler Reports, OLAP Reports, TimesTen Reports, User Defined Reports.
- SSH Hosts:** SSH Hosts.
- REST Development:** REST Data Services.
- Worksheet:** The main workspace displays the executed SQL script. The script creates several types and a table:
 

```

member function containsText (pString1 varchar2, pString2 varchar2) return integer
not instantiable not final
/
create or replace type disk_type under album_type
( mediaType      varchar(10),
diskNum         number(2), -- number of disks
diskUsedPrice   number(9,2),
diskDeliveryCost number(9,2),
overriding member function discountPrice return number)
/
create or replace type mp3_type under album_type
(downloadSize number, -- size in MB
overriding member function discountPrice return number)
/
-- create tables --
create table albums of album_type
object id system generated
nested table albumReviews store as store_reviews
/
      
```
- Script Output:** Task completed in 2.473 seconds.
- Messages - Log:** Shows compilation messages for various types:
  - Type ARTIST\_TYPE compiled
  - Type ARTIST\_ARRAY\_TYPE compiled
  - Type REVIEW\_TYPE compiled
  - Type REVIEW\_TABLE\_TYPE compiled
  - Type ALBUM\_TYPE compiled
  - Type DISK\_TYPE compiled
  - Type MP3\_TYPE compiled
- Table:** ALBUMS created.

->A script of INSERT statements to populate the OMDB database using the data supplied in the `omdb_data.txt` file and suitable constructor statements.

-----**INSERTION**-----

-----**Vinyl Disk**-----

```

INSERT INTO albums
values (disk_type('The Essential Bob Dylan',99,'8-
Jul-2016','Pop',37.00,32,
artist_array_type(
    artist_type('Bob Dylan','Composer'),
    artist_type('Bob Dylan','Vocals')
),
review_table_type(
    review_type('Shawn','24-Jul-2018',' Wife loved it!',5),
    review_type('Reuben','2-
Aug-2019','Great compilation of some of his most known songs',5)
),
'Vinyl', 2, null,11.00));

INSERT INTO albums
values (disk_type('Sketches of Spain',45,'8-Mar-2011','Jazz',14.99,6,
artist_array_type(
    artist_type('Miles Davis','Composer'),
    artist_type('Miles Davis','Musician')
),
review_table_type(
    review_type('Frederick','16-
Sep-2016','Recommend listening while viewing a sunset',5),
    review_type('Juliet','12-Mar-2018','Early days of the greatest Miles--no lover of jazz should be without this album',5)
),
'Vinyl', 1, 16.29, 7.00));

INSERT INTO albums
values (disk_type('Bob Dylan's Greatest Hits',45, '31-
Jan-2017','Pop Rock',29.87,10,
artist_array_type(
    artist_type('Bob Dylan','Composer'),
    artist_type('Bob Dylan','Vocals'))
)

```

```

),
review_table_type(
    review_type('Kandy','16-Mar-2015','Early Dylan in all his glory.',5),
    review_type('Stewart','18-
Feb-2013','Captures Bob Dylan transformation from a folk song Composer to
a rock legend',4)
),
'Vinyl', 1, null, 11.00));

INSERT INTO albums
values (disk_type('Harvest (2009 Remaster)',44,'21-
Jun-2009','Rock Country',28.50,10,
artist_array_type(
    artist_type('Neil Young','Composer'),
    artist_type('Neil Young','Vocals')
),
review_table_type(
    review_type('John','18-Feb-2019','I absolutely LOVE this CD!',5),
    review_type('Stewart','18-Feb-2013','Sounds good on vinyl!',5)
),
'Vinyl', 1, 14.99, 11.00));

INSERT INTO albums
values (disk_type('Kind Of Blue (Legacy Edition)',155,'20-
Jan-2009','Jazz',19.99,21,
artist_array_type(
    artist_type('Miles Davis','Composer'),
    artist_type('Miles Davis','Musician')
),
review_table_type(
    review_type('Laurence','10-Sep-2014','Very very special recording.',5)
),
'Vinyl', 3, 16.99, 10.00));

```

#### -----Audio CDs-----

```

INSERT INTO albums
values (disk_type('Harvest (2009 Remaster)',44,'21-
Jun-2009','Rock Country',10.50, 10,
artist_array_type(

```

```

    artist_type('Neil Young', 'Composer'),
    artist_type('Neil Young', 'Vocals')
),
review_table_type(
    review_type('John', '18-Feb-2019', ' I absolutely LOVE this CD!', 5),
    review_type('Anthony', '16-Aug-2019', 'Neil Youngs signature album.', 4)
),
'Audio CD', 1, 4.99, 11.00));

INSERT INTO albums
values (disk_type('The Essential Bob Dylan', 99, '8-
Jul-2016', 'Pop', 26.17, 32,
artist_array_type(
    artist_type('Bob Dylan', 'Composer'),
    artist_type('Bob Dylan', 'Vocals')
),
review_table_type(
    review_type('Christopher', '24-
Jun-2016', 'This is a terrific album.', 5),
    review_type('Cauley', '2-
Aug-2015', 'There can only be one Bob Dylan. God blessed him with the gift
of verse.', 5)
),
'Audio CD', 2, null, 7.00));

INSERT INTO albums
values (disk_type('Bob Dylans Greatest Hits', 50, '1-
Jun-1999', 'Pop Rock', 20.81, 10,
artist_array_type(
    artist_type('Bob Dylan', 'Composer'),
    artist_type('Bob Dylan', 'Vocals')
),
review_table_type(
    review_type('Kandy', '16-Mar-2015', 'Early Dylan in all his glory.', 5),
    review_type('Stewart', '18-
Feb-2013', 'Captures Bob Dylan transformation from a folk song Composer to
a rock legend', 4)
),
'Audio CD', 1, null, 7.00));

```

```

INSERT INTO albums
values (disk_type('Kind Of Blue (Legacy Edition)',155,'20-
Jan-2009','Jazz', 19.99, 21,
artist_array_type(
    artist_type('Miles Davis','Composer'),
    artist_type('Miles Davis','Musician')
),
review_table_type(
    review_type('Amy','17-
Apr-2018','Poor quality sound compared to the vinyl record.',2)
),
'Audio CD', 3, 16.99, 10.00));

INSERT INTO albums
values (disk_type('Sketches of Spain', 45,'20-Jan-2009','Jazz', 3.11, 6,
artist_array_type(
    artist_type('Miles Davis','Composer'),
    artist_type('Miles Davis','Musician')
),
review_table_type(
    review_type('Sara','3-
Oct-2016','Another Must Have! One of Miles finest works.',5),
    review_type('Douglas','14-
Jun-2014','You might like it, but I admit it seems like a difficult listen
.',5)
),
'Audio CD', 1, 6.41, 7.00));

INSERT INTO albums
values (disk_type('Gustav Mahler Symphony No. 9',45,'12-
Oct-2017','Classical', 23.10, 5,
artist_array_type(
    artist_type('David Zinman','Conductor'),
    artist_type('Gustav Mahler','Composer'),
    artist_type('Tonhalle Orchestra','Orchestra')
),
review_table_type(
    review_type('Lindon','3-
Dec-2010','This is an uneventful but fine recording.',3),

```

```

review_type('Prescott', '24-
Aug-2013', 'This is truly a spellbinding record.', 5)
),
'Audio CD', 1, 15.20, 7.00));

```

-----MP3-----

```

INSERT INTO albums
values (mp3_type('Bob Dylans Greatest Hits', 55, '1-
Jan-2019', 'Pop Rock', 5.98, 10,
artist_array_type(
    artist_type('Bob Dylan', 'Composer'),
    artist_type('Bob Dylan', 'Vocals')
),
review_table_type(
    review_type('Mandy', '16-Mar-2019', 'Fantastic music!', 5)
),
60));

INSERT INTO albums
values (mp3_type('Best of Neil Young', 153, '21-
Feb-2019', 'Pop Rock', 17.50, 35,
artist_array_type(
    artist_type('Neil Young', 'Composer'),
    artist_type('Neil Young', 'Vocals')
),
review_table_type(
    review_type('John', '16-Apr-2019', 'Great artist and great music.', 5)
),
165));

INSERT INTO albums
values (mp3_type('Harvest (2009 Remaster)', 44, '21-
Jun-2009', 'Rock Country', 9.49, 10,
artist_array_type(
    artist_type('Neil Young', 'Composer'),
    artist_type('Neil Young', 'Vocals')
),
review_table_type(

```

```

    review_type('John', '16-Apr-2019', 'Great artist and great music.', 5)
),
52));

INSERT INTO albums
values (mp3_type('Sketches of Spain', 45, '16-Aug-2013', 'Jazz', 24.99, 6,
artist_array_type(
    artist_type('Miles Davis', 'Composer'),
    artist_type('Miles Davis', 'Musician')
),
review_table_type(
    review_type('Douglas', '14-
Jun-2014', 'You might like it but I admit it seems like a difficult listen.
', 5)
),
51));

INSERT INTO albums
values (mp3_type('B.B. King Greatest Hits', 114, '16-
Jul-2013', 'Rock Blues', 11.49, 24,
artist_array_type(
    artist_type('B.B. King', 'Vocals'),
    artist_type('B.B. King', 'Guitar')
),
review_table_type(
    review_type('David', '18-
May-2015', 'I highly recommend this album to anyone who want to see what BB
King is all about.', 4)
),
125));

INSERT INTO albums
values (mp3_type('The Essential Bob Dylan', 99, '8-
Jul-2016', 'Pop', 16.00, 32,
artist_array_type(
    artist_type('Bob Dylan', 'Composer'),
    artist_type('Bob Dylan', 'Vocals')
),
review_table_type(

```

```

review_type('Christopher', '24-Jun-2016', 'This is a terrific album.', 5),
review_type('Cauley', '2-Apr-2015', 'There can only be one Bob Dylan. God blessed him with the gift of verse', 5)
),
112));

```

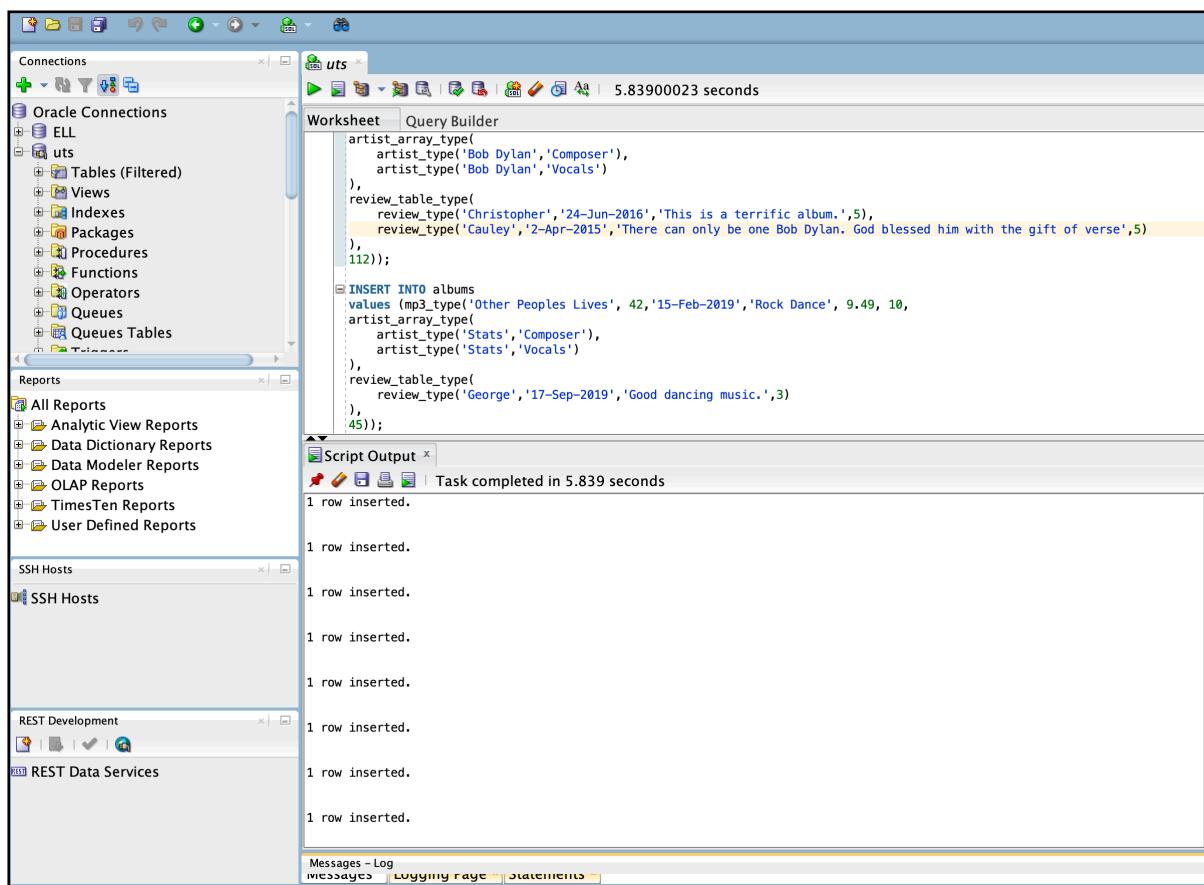
  

```

INSERT INTO albums
values (mp3_type('Other Peoples Lives', 42, '15-Feb-2019', 'Rock Dance', 9.49, 10,
artist_array_type(
    artist_type('Stats', 'Composer'),
    artist_type('Stats', 'Vocals')
),
review_table_type(
    review_type('George', '17-Sep-2019', 'Good dancing music.', 3)
),
45));

```

### \*Results of the script(insert) execution:



The screenshot shows the Oracle SQL Developer interface. The left sidebar contains connections to 'Oracle Connections' (with 'ELL' and 'uts' selected), 'Reports' (including 'All Reports', 'Analytic View Reports', etc.), and 'SSH Hosts'. The main area has tabs for 'Worksheet' and 'Query Builder'. The 'Worksheet' tab displays the following SQL script:

```

artist_array_type(
    artist_type('Bob Dylan', 'Composer'),
    artist_type('Bob Dylan', 'Vocals')
),
review_table_type(
    review_type('Christopher', '24-Jun-2016', 'This is a terrific album.', 5),
    review_type('Cauley', '2-Apr-2015', 'There can only be one Bob Dylan. God blessed him with the gift of verse', 5)
),
112));

INSERT INTO albums
values (mp3_type('Other Peoples Lives', 42, '15-Feb-2019', 'Rock Dance', 9.49, 10,
artist_array_type(
    artist_type('Stats', 'Composer'),
    artist_type('Stats', 'Vocals')
),
review_table_type(
    review_type('George', '17-Sep-2019', 'Good dancing music.', 3)
),
45));

```

The 'Script Output' tab shows the execution results:

```

1 row inserted.

```