

## Database Design Project

### Oracle Baseball League Store Database

#### Project Scenario:

You are a small consulting company specializing in database development. You have just been awarded the contract to develop a data model for a database application system for a small retail store called Oracle Baseball League (OBL).

The Oracle Baseball League store serves the entire surrounding community selling baseball kit. The OBL has two types of customer, there are individuals who purchase items like balls, cleats, gloves, shirts, screen printed t-shirts, and shorts. Additionally customers can represent a team when they purchase uniforms and equipment on behalf of the team.

Teams and individual customers are free to purchase any item from the inventory list, but teams get a discount on the list price depending on the number of players. When a customer places an order we record the order items for that order in our database.

OBL has a team of three sales representatives that officially only call on teams but have been known to handle individual customer complaints.

#### Section 6 Lesson 9 Exercise 1: Joining Tables Using JOIN

##### Write SELECT Statements Using Data From Multiple Tables Using Equijoins and Non-Equijoins (S6L9 Objective 1)

In this exercise you will write SELECT statements to access data from more than one table.

## Part 1: Creating Natural Joins.

1. Display all of the information about sales representatives and their addresses using a natural join.

The screenshot shows the APEX SQL Workshop interface. The SQL command entered is:

```
1 SELECT *
2 FROM sales_representatives NATURAL JOIN sales_rep_addresses
```

The results are displayed in a table with 11 columns: ID, EMAIL, FIRST\_NAME, LAST\_NAME, PHONE\_NUMBER, COMMISSION\_RATE, SUPERVISOR\_ID, ADDRESS\_LINE\_1, ADDRESS\_LINE\_2, CITY, and ZIP\_CODE. The table contains 3 rows of data.

ID	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	COMMISSION_RATE	SUPERVISOR_ID	ADDRESS_LINE_1	ADDRESS_LINE_2	CITY	ZIP_CODE
sr01	chray@obl.com	Charles	Raymond	0134598761	10	sr01	12 Cherry Lane	Denton	Detroit	DT48211
sr02	vwright@obl.com	Victoria	Wright	0134598762	5	sr01	87 Blossom Hill	Uptown	Detroit	DT52314
sr03	bspeed@obl.com	Barry	Speed	0134598763	5	sr01	12 Junction Row	Skinflats	Detroit	DT52564

3 rows returned in 0.01 seconds

2. Adapt the query from the previous question to only show the id, first name, last name, address line 1, address line 2, city, email and phone\_number for the sales representatives.

The screenshot shows the APEX SQL Workshop interface. The SQL command entered is:

```
1 SELECT id, first_name, last_name, address_line_1, address_line_2, city, email, phone_number
2 FROM sales_representatives NATURAL JOIN sales_rep_addresses
```

The results are displayed in a table with 8 columns: ID, FIRST\_NAME, LAST\_NAME, ADDRESS\_LINE\_1, ADDRESS\_LINE\_2, CITY, EMAIL, and PHONE\_NUMBER. The table contains 3 rows of data.

ID	FIRST_NAME	LAST_NAME	ADDRESS_LINE_1	ADDRESS_LINE_2	CITY	EMAIL	PHONE_NUMBER
sr01	Charles	Raymond	12 Cherry Lane	Denton	Detroit	chray@obl.com	0134598761
sr02	Victoria	Wright	87 Blossom Hill	Uptown	Detroit	vwright@obl.com	0134598762
sr03	Barry	Speed	12 Junction Row	Skinflats	Detroit	bspeed@obl.com	0134598763

3 rows returned in 0.00 seconds

## Part 2: Creating Joins with the USING Clause

1. Adapt the previous query answer to use the USING clause instead of a natural join.

The screenshot shows the APEX SQL Workshop interface. The query editor contains the following SQL code:

```
1 SELECT id, first_name, last_name, address_line_1, address_line_2, city, email, phone_number
2 FROM sales_representatives JOIN sales_rep_addresses
3 USING (id);
```

The results tab displays the following data:

ID	FIRST_NAME	LAST_NAME	ADDRESS_LINE_1	ADDRESS_LINE_2	CITY	EMAIL	PHONE_NUMBER
sr01	Charles	Raymond	12 Cherry Lane	Denton	Detroit	chray@obl.com	0134598761
sr02	Victoria	Wright	87 Blossom Hill	Uptown	Detroit	vwright@obl.com	0134598762
sr03	Barry	Speed	12 Junction Row	Skinflats	Detroit	bspeed@obl.com	0134598763

3 rows returned in 0.05 seconds

2. Display all of the information about items and their price history by joining the items and price\_history tables.

The screenshot shows the APEX SQL Workshop interface. The query editor contains the following SQL code:

```
1 SELECT *
2 FROM items JOIN price_history
3 USING (itm_number);
```

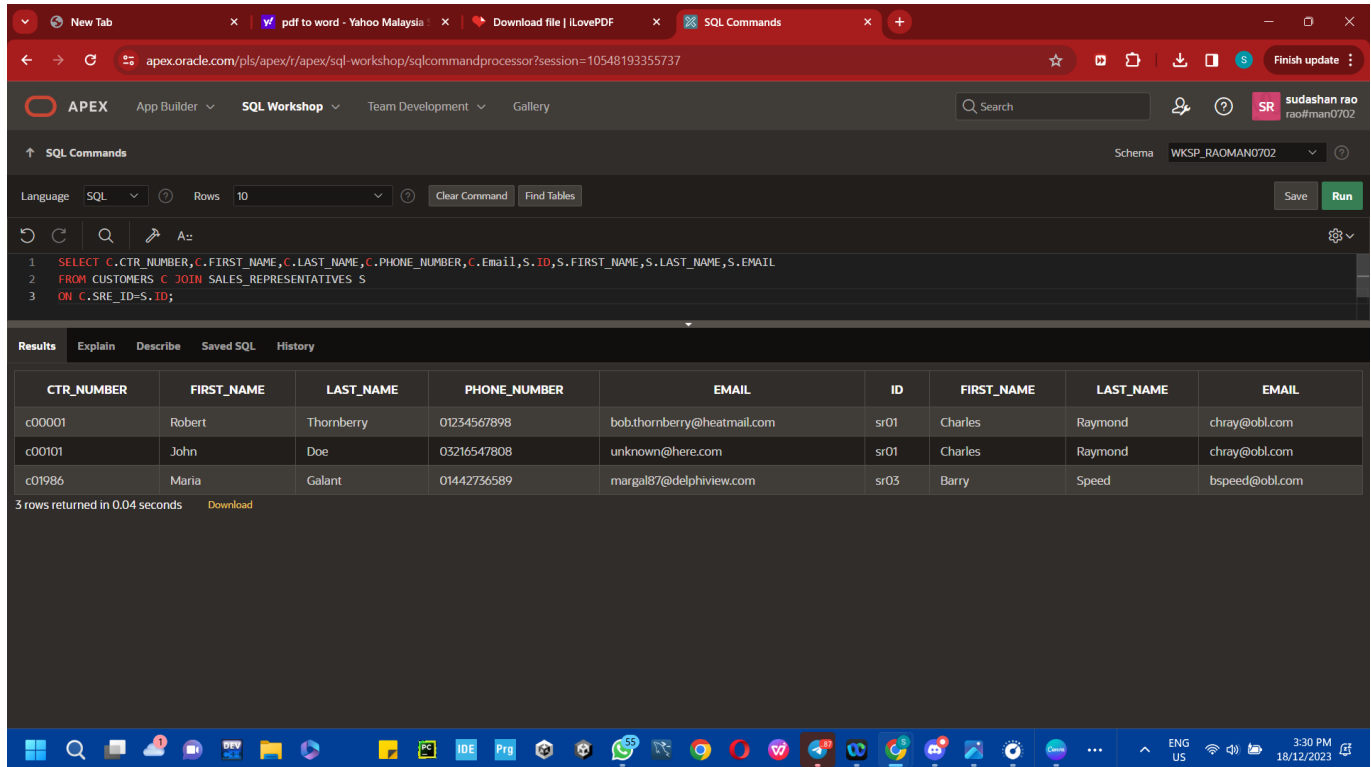
The results tab displays the following data:

ITM_NUMBER	NAME	DESCRIPTION	CATEGORY	COLOR	Size	ILT_ID	START_DATE	START_TIME	PRICE	END_DATE	END_TIME
im01101044	gloves	catcher mitt	clothing	brown	m	il010230124	06/17/2017	06/17/2016	4.99	-	-
im01101045	under shirt	top worn under the game top	clothing	white	s	il010230125	11/25/2016	11/25/2016	14.99	01/25/2017	01/25/2017
im01101045	under shirt	top worn under the game top	clothing	white	s	il010230125	01/25/2017	01/25/2017	8.99	01/25/2017	01/25/2017
im01101045	under shirt	top worn under the game top	clothing	white	s	il010230125	01/26/2017	01/26/2017	15.99	-	-
im01101046	socks	team socks with emblem	clothing	range	l	il010230126	02/12/2017	02/12/2017	7.99	-	-
im01101047	game top	team shirt with emblem	clothing	range	m	il010230127	04/25/2017	04/25/2017	24.99	-	-
im01101048	premium bat	high quality baseball bat	equipment	-	-	il010230128	05/31/2017	05/31/2017	149	11/07/2023	11/07/2023
im01101048	premium bat	high quality baseball bat	equipment	-	-	il010230128	11/07/2023	11/07/2023	99.99	-	-

8 rows returned in 0.04 seconds

### Part 3: Creating Joins with the ON Clause

1. Use an ON clause to join the customer and sales representative table so that you display the customer number, customer first name, customer last name, customer phone number, customer email, sales representative id, sales representative first name, sales representative last name and sales representative email. You will need to use a table alias in your answer as both tables have columns with the same name.



The screenshot shows the APEX SQL Workshop interface. The SQL command entered is:

```
1 SELECT C.CTR_NUMBER,C.FIRST_NAME,C.LAST_NAME,C.PHONE_NUMBER,C.Email,S.ID,S.FIRST_NAME,S.LAST_NAME,S.EMAIL
2 FROM CUSTOMERS C JOIN SALES_REPRESENTATIVES S
3 ON C.SRE_ID=S.ID;
```

The results table shows the following data:

CTR_NUMBER	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	ID	FIRST_NAME	LAST_NAME	EMAIL
c00001	Robert	Thornberry	01234567898	bob.thornberry@heatmail.com	sr01	Charles	Raymond	chray@obl.com
c00101	John	Doe	03216547808	unknown@here.com	sr01	Charles	Raymond	chray@obl.com
c01986	Maria	Galant	01442736589	margal87@delphiview.com	sr03	Barry	Speed	bspeed@obl.com

3 rows returned in 0.04 seconds

## Part 4- Creating Three-Way Joins with the ON Clause

1. Using the answer to Task 3 add a join that will allow the team name that the customer represents to be included in the results.

The screenshot shows the APEX SQL Workshop interface. The SQL command editor contains the following query:

```
1 SELECT C.CTR_NUMBER,C.FIRST_NAME,C.LAST_NAME,C.PHONE_NUMBER,C.Email,S.ID,S.FIRST_NAME,S.LAST_NAME,S.EMAIL, T.NAME AS "TEAM NAME"
2 FROM CUSTOMERS C JOIN SALES_REPRESENTATIVES S
3 ON C.SRE_ID=S.ID
4 JOIN TEAMS T
5 ON C.TEM_ID=T.ID;
```

The Results tab shows the following data:

CTR_NUMBER	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	ID	FIRST_NAME	LAST_NAME	EMAIL	TEAM NAME
c00001	Robert	Thornberry	01234567898	bob.thornberry@heatmail.com	sr01	Charles	Raymond	chray@obl.com	Rockets
c00101	John	Doe	03216547808	unknown@here.com	sr01	Charles	Raymond	chray@obl.com	Celtics
c01986	Maria	Galanit	01442736589	margal87@delphiview.com	sr03	Barry	Speed	bspeed@obl.com	Rovers

3 rows returned in 0.04 seconds

## Part 5: Applying Additional Conditions to a Join

- Using the answer to Task 4 add an additional condition to only show the results for the customer that has the number - c00001.

The screenshot shows the APEX SQL Workshop interface. The SQL command area contains the following query:

```
1 SELECT C.CTR_NUMBER,C.FIRST_NAME,C.LAST_NAME,C.PHONE_NUMBER,C.Email,S.ID,S.FIRST_NAME,S.LAST_NAME,S.EMAIL, T.NAME AS "TEAM NAME"
2 FROM CUSTOMERS C JOIN SALES_REPRESENTATIVES S
3 ON C.SRE_ID=S.ID
4 JOIN TEAMS T
5 ON C.TEM_ID=T.ID
6 WHERE CTR_NUMBER='c00001';
```

The Results tab shows the following data:

CTR_NUMBER	FIRST_NAME	LAST_NAME	PHONE_NUMBER	EMAIL	ID	FIRST_NAME	LAST_NAME	EMAIL	TEAM NAME
c00001	Robert	Thornberry	01234567898	bob.thornberry@heatmail.com	sr01	Charles	Raymond	chray@obl.com	Rockets

1 rows returned in 0.03 seconds

## Part 6: Retrieving Records with Nonequijoins

- Write a query that will display name and cost of the item with the number im01101045 on the 12<sup>th</sup> of December 2016. The output of the query should look like this:

The cost of the under shirt on this day was 14.99

The screenshot shows the APEX SQL Workshop interface. The SQL command area contains the following query:

```
1 SELECT 'The cost of the '|| I.NAME || 'on this day was'|| P.PRICE AS "ITEM DETAILS"
2 FROM ITEMS I
3 JOIN PRICE_HISTORY P ON (TO_DATE('12-Dec-2016','DD-Mon-YYYY')BETWEEN P.START_DATE AND P.END_DATE)
4 WHERE I.ITHM_NUMBER='im01101045';
```

The Results tab shows the following data:

ITEM DETAILS
The cost of the under shirton this day was14.99

1 rows returned in 0.01 seconds

Copyright © 2020, Oracle and/or its affiliates. All rights reserved. Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.