



SECD2523 – DATABASE

SEMESTER 1/20232024

SECTION 10

LAB 4: DML 3 PART 1

NAME: LIM XIAO XUAN

MATRIC NO: A22EC0071

LECTURER: DR. ROZILAWATI BINTI DOLLAH@MD.ZAIN

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.

```
SELECT *  
FROM sales_representatives  
NATURAL JOIN sales_rep_addresses;
```

| 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 |

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.

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

| 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 |

Part 2: Creating Joins with the USING Clause

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

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

| 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 |

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

```
SELECT *
FROM items
JOIN price_history ON items.itm_number = price_history.itm_number;
```

| ITM_NUMBER | NAME | DESCRIPTION | CATEGORY | COLOR | Size | ILT_ID | START_DATE | START_TIME | PRICE | END_DATE | END_TIME | ITM_NUMBER |
|------------|-------------|-----------------------------|-----------|-------|------|-------------|------------|------------|-------|-----------|-----------|------------|
| im01101044 | gloves | catcher mitt | clothing | brown | m | il010230124 | 17-JUN-17 | 17-JUN-16 | 4.99 | - | - | im01101044 |
| im01101045 | under shirt | top worn under the game top | clothing | white | s | il010230125 | 25-NOV-16 | 25-NOV-16 | 14.99 | 25-JAN-17 | 25-JAN-17 | im01101045 |
| im01101045 | under shirt | top worn under the game top | clothing | white | s | il010230125 | 25-JAN-17 | 25-JAN-17 | 8.99 | 25-JAN-17 | 25-JAN-17 | im01101045 |
| im01101045 | under shirt | top worn under the game top | clothing | white | s | il010230125 | 26-JAN-17 | 26-JAN-17 | 15.99 | - | - | im01101045 |
| im01101046 | socks | team socks with emblem | clothing | range | l | il010230126 | 12-FEB-17 | 12-FEB-17 | 7.99 | - | - | im01101046 |
| im01101047 | game top | team shirt with emblem | clothing | range | m | il010230127 | 25-APR-17 | 25-APR-17 | 24.99 | - | - | im01101047 |
| im01101048 | premium bat | high quaity baseball bat | equipment | - | - | il010230128 | 31-MAY-17 | 31-MAY-17 | 149 | 22-DEC-23 | 22-DEC-23 | im01101048 |
| im01101048 | premium bat | high quaity baseball bat | equipment | - | - | il010230128 | 22-DEC-23 | 22-DEC-23 | 99.99 | - | - | im01101048 |

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.

```
SELECT c.ctr_number,  
       c.first_name AS "Customer First Name",  
       c.last_name AS "Customer Last Name",  
       c.phone_number AS "Customer Phone Number",  
       c.email AS "Customer Email",  
       s.id AS "Sales Representative ID",  
       s.first_name AS "Sales Representative First Name",  
       s.last_name AS "Sales Representative Last Name",  
       s.email AS "Sales Representative Email"  
FROM customers c  
JOIN sales_representatives s ON c.sre_id = s.id;
```

| CTR_NUMBER | Customer First Name | Customer Last Name | Customer Phone Number | Customer Email | Sales Representative ID | Sales Representative First Name | Sales Representative Last Name | Sales Representative 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 |

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.

```
SELECT
    c.ctr_number,
    c.first_name AS "Customer First Name",
    c.last_name AS "Customer Last Name",
    c.phone_number AS "Customer Phone Number",
    c.email AS "Customer Email",
    s.id AS "Sales Representative ID",
    s.first_name AS "Sales Representative First Name",
    s.last_name AS "Sales Representative Last Name",
    s.email AS "Sales Representative Email",
    t.name AS "Team Name"
FROM customers c
JOIN sales_representatives s ON c.sre_id = s.id
JOIN teams t ON c.tem_id = t.id;
JOIN sales_representatives s ON c.sre_id = s.id
JOIN teams t ON c.tem_id = t.id;
```

| CTR_NUMBER | Customer First Name | Customer Last Name | Customer Phone Number | Customer Email | Sales Representative ID | Sales Representative First Name | Sales Representative Last Name | Sales Representative 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 | Galant | 01442736589 | margal87@delphiview.com | sr03 | Barry | Speed | bspeed@obl.com | Rovers |

Part 5: Applying Additional Conditions to a Join

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

```
SELECT
  c.ctr_number,
  c.first_name AS "Customer First Name",
  c.last_name AS "Customer Last Name",
  c.phone_number AS "Customer Phone Number",
  c.email AS "Customer Email",
  s.id AS "Sales Representative ID",
  s.first_name AS "Sales Representative First Name",
  s.last_name AS "Sales Representative Last Name",
  s.email AS "Sales Representative Email",
  t.name AS "Team Name"
FROM customers c
JOIN sales_representatives s ON c.sre_id = s.id
JOIN teams t ON c.tem_id = t.id
WHERE c.ctr_number = 'c00001';
```

| CTR_NUMBER | Customer First Name | Customer Last Name | Customer Phone Number | Customer Email | Sales Representative ID | Sales Representative First Name | Sales Representative Last Name | Sales Representative Email | Team Name |
|------------|---------------------|--------------------|-----------------------|-----------------------------|-------------------------|---------------------------------|--------------------------------|----------------------------|-----------|
| c00001 | Robert | Thornberry | 01234567898 | bob.thornberry@heatmail.com | sr01 | Charles | Raymond | chray@obl.com | Rockets |

Part 6: Retrieving Records with Nonequijoins

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

The cost of the under shirt on this day was 14.99

```
SELECT 'The cost of the ' || i.name || ' on this day was ' || ph.price AS output
FROM items i
LEFT JOIN price_history ph ON i.itm_number = ph.itm_number
WHERE i.itm_number = 'im01101045'
AND TO_DATE('12-Dec-2016', 'DD-Mon-YYYY') BETWEEN ph.start_date AND NVL(ph.end_date,
TO_DATE('31-Dec-9999', 'DD-Mon-YYYY'));
```

OUTPUT

```
The cost of the under shirt on this day was 14.99
```