

SECD2523 – DATABASE

SEMESTER 1/20232024

SECTION 08

LAB 4: DML3

LECTURER: DR. NOOR HIDAYAH BINTI ZAKARIA

NAME	MATRIC NO.
EILEEN YONG KAI QIN	A22EC0048

Section 6 Lesson 9 Exercise 1: Joining Tables Using JOIN

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;



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 s

NATURAL JOIN sales rep addresses a;



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);



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;



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 fist 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;



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
```



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",

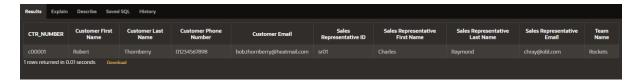
t.name AS "Team Name"

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';
```



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'));



Section 6 Lesson 9 Exercise 2: Joining Tables Using JOIN

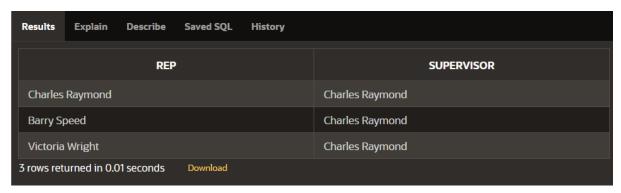
Part 1: Use a Self-Join to Join a Table to Itself (S6L9 Objective 2)

1. Write a query that will display who the supervisor is for each of the sales representatives. The information should be displayed in two columns, the first column will be the first name and last name of the sales representative and the second will be the first name and last name of the supervisor. The column aliases should be Rep and Supervisor.

SELECT s1.first_name || ' ' || s1.last_name AS Rep, s2.first_name || ' ' || s2.last_name AS Supervisor

FROM sales representatives s1

JOIN sales_representatives s2 ON s1.supervisor_id = s2.id;



Part 2: Use OUTER joins (S6L9 Objective 3)

1. Write a query that will display all of the team and customer information even if there is no match with the table on the left (team).

SELECT t.*, c.*

FROM teams t

LEFT JOIN customers c ON t.id = c.tem id;



Part 3: Generating a Cartesian Product (S6L9 Objective 4)

1. Create a Cartesian product between the customer and sales representative tables.

SELECT*

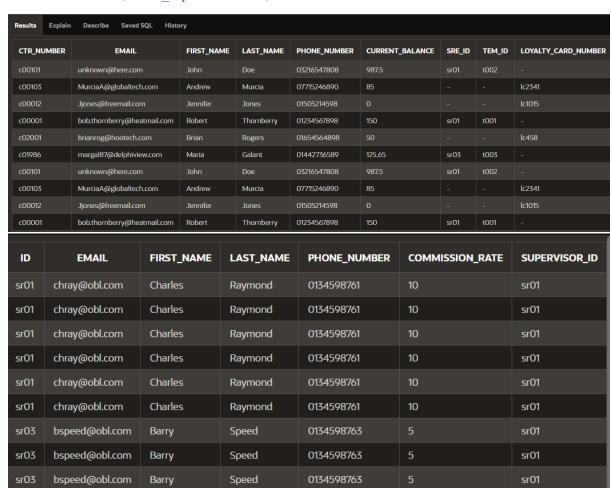
sr03

bspeed@obl.com

Barry

Speed

FROM customers, sales representatives;



0134598763

sr01