



SECD2523 – DATABASE

SEMESTER 1/20232024

SECTION 08

LAB 4: DML3

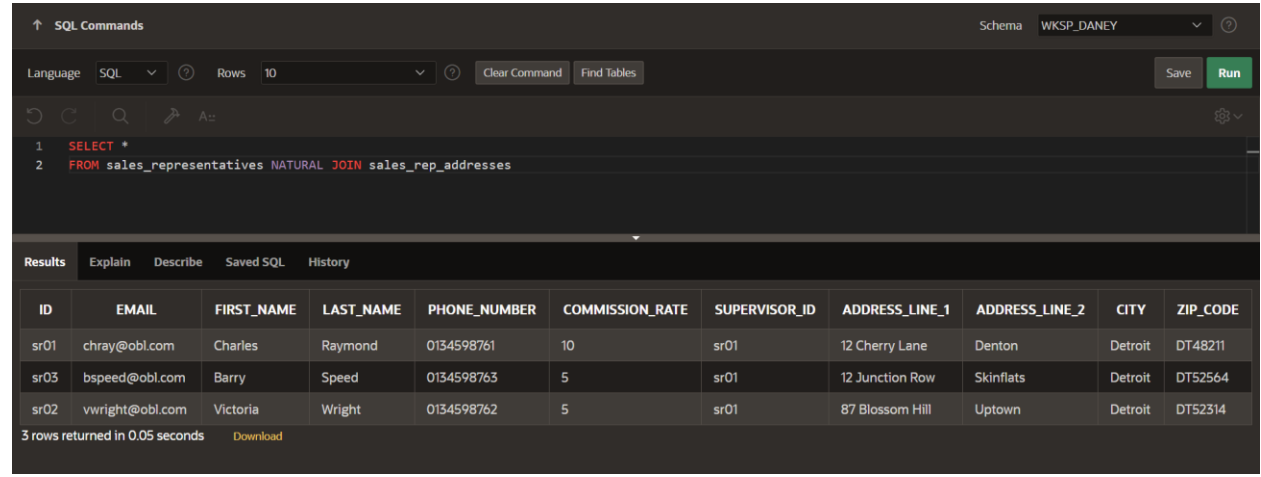
LECTURER: DR. NOOR HIDAYAH BINTI ZAKARIA

NAME	MATRIC NUMBER
<i>ABDULRAHMAN ABDULLAH AHMED DANAY</i>	<i>A22EC4001</i>

## PART 1:

### Creating Natural Joins:

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



The screenshot shows a SQL IDE interface with the following components:

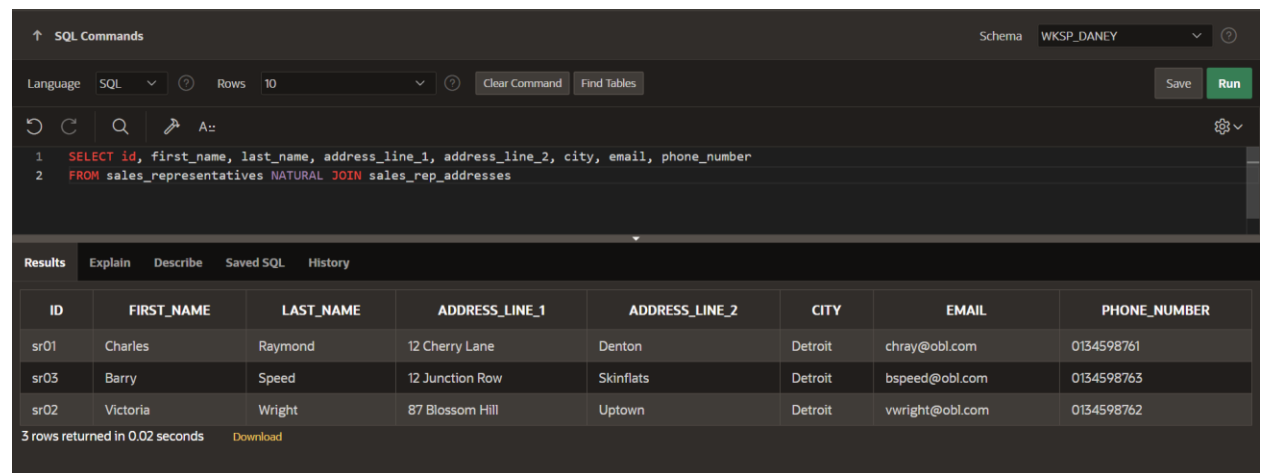
- SQL Commands:** Language: SQL, Rows: 10. Buttons: Clear Command, Find Tables, Save, Run.
- Query:**

```
1 SELECT *
2 FROM sales_representatives NATURAL JOIN sales_rep_addresses
```
- Results:** Explain, Describe, Saved SQL, History.
- Table:**

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
sr03	bspeed@obl.com	Barry	Speed	0134598763	5	sr01	12 Junction Row	Skinflats	Detroit	DT52564
sr02	vwright@obl.com	Victoria	Wright	0134598762	5	sr01	87 Blossom Hill	Uptown	Detroit	DT52314

3 rows returned in 0.05 seconds Download

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 a SQL IDE interface with the following components:

- SQL Commands:** Language: SQL, Rows: 10. Buttons: Clear Command, Find Tables, Save, Run.
- Query:**

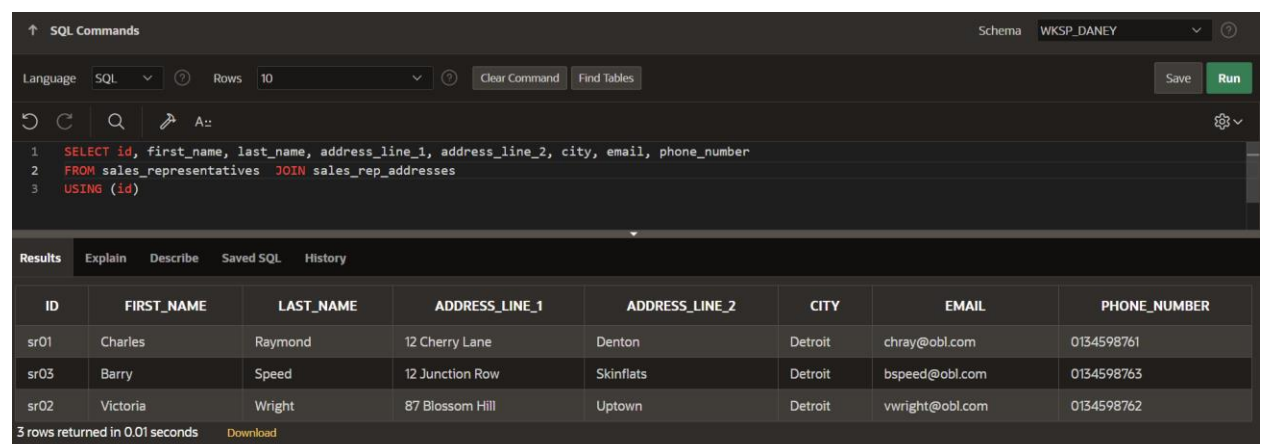
```
1 SELECT id, first_name, last_name, address_line_1, address_line_2, city, email, phone_number
2 FROM sales_representatives NATURAL JOIN sales_rep_addresses
```
- Results:** Explain, Describe, Saved SQL, History.
- Table:**

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
sr03	Barry	Speed	12 Junction Row	Skinflats	Detroit	bspeed@obl.com	0134598763
sr02	Victoria	Wright	87 Blossom Hill	Uptown	Detroit	vwright@obl.com	0134598762

3 rows returned in 0.02 seconds Download

### 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 a SQL IDE interface with the following components:

- SQL Commands:** Language: SQL, Rows: 10. Buttons: Clear Command, Find Tables, Save, Run.
- Query:**

```
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)
```
- Results:** Explain, Describe, Saved SQL, History.
- Table:**

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
sr03	Barry	Speed	12 Junction Row	Skinflats	Detroit	bspeed@obl.com	0134598763
sr02	Victoria	Wright	87 Blossom Hill	Uptown	Detroit	vwright@obl.com	0134598762

3 rows returned in 0.01 seconds Download

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

SQL Commands

Language: SQL Rows: 10

```

1 SELECT *
2 FROM items JOIN price_history
3 USING (itm_number)
4
5

```

Results

ITM_NUMBER	NAME	DESCRIPTION	CATEGORY	COLOR	Size	ILT_ID	START_DATE	START_TIME	PRICE	END_DATE	END_TIME
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	-	-
im01101047	game top	team shirt with emblem	clothing	range	m	il010230127	04/25/2017	04/25/2017	24.99	-	-
im01101044	gloves	catcher mitt	clothing	brown	m	il010230124	06/17/2017	06/17/2016	4.99	-	-

## 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.

SQL Commands

Language: SQL Rows: 10

```

1 SELECT c.ctr_number "Customer Number", c.first_name "Customer First Name", c.last_name "Customer Last Name", c.phone_number "Customer Phone Number",
2 c.email "Customer Email",
3 s.id "Sales Representatives ID", s.first_name "Sales Representatives First Name", s.last_name, s.email "Sales Representatives Last Name"
4 FROM customers c JOIN sales_representatives s
5 ON (c.sre_id = s.id)
6

```

Results

Customer Number	Customer First Name	Customer Last Name	Customer Phone Number	Customer Email	Sales Representatives ID	Sales Representatives First Name	LAST_NAME	Sales Representatives Last Name
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 0.03 seconds

## 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 SQL Developer interface with the following SQL query:

```
1 SELECT c.ctr_number "Customer Number", c.first_name "Customer First Name", c.last_name "Customer Last Name", c.phone_number "Customer Phone Number",  
2 c.email "Customer Email",  
3 s.id "Sales Representatives ID", s.first_name "Sales Representatives First Name", s.last_name, s.email "Sales Representatives Last Name"  
4 FROM customers c JOIN sales_representatives s  
5 ON (c.sre_id = s.id)  
6 JOIN teams t  
7 ON (c.tem_id = t.id)
```

The results table displays the following data:

Customer Number	Customer First Name	Customer Last Name	Customer Phone Number	Customer Email	Sales Representatives ID	Sales Representatives First Name	LAST_NAME	Sales Representatives Last Name
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

## 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.

The screenshot shows the SQL Developer interface with the following SQL query:

```
1 SELECT c.ctr_number "Customer Number", c.first_name "Customer First Name", c.last_name "Customer Last Name", c.phone_number "Customer Phone Number",  
2 c.email "Customer Email",  
3 s.id "Sales Representatives ID", s.first_name "Sales Representatives First Name", s.last_name, s.email "Sales Representatives Last Name"  
4 FROM customers c JOIN sales_representatives s  
5 ON (c.sre_id = s.id)  
6 JOIN teams t  
7 ON (c.tem_id = t.id)  
8 where c.ctr_number = 'c00001'
```

The results table displays the following data:

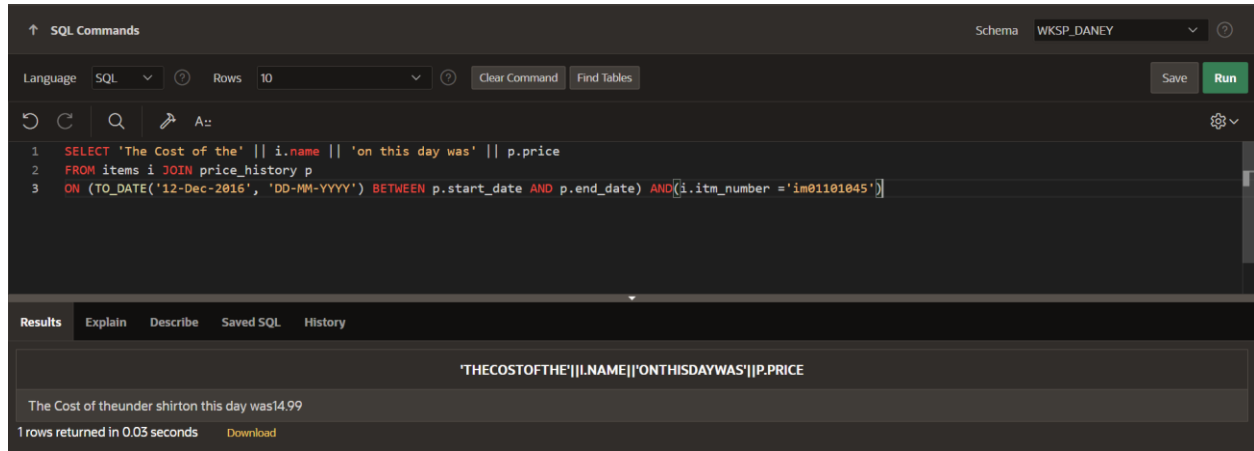
Customer Number	Customer First Name	Customer Last Name	Customer Phone Number	Customer Email	Sales Representatives ID	Sales Representatives First Name	LAST_NAME	Sales Representatives Last Name
c00001	Robert	Thornberry	01234567898	bob.thornberry@heatmail.com	sr01	Charles	Raymond	chray@obl.com

1 rows returned in 0.01 seconds

## 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



The screenshot shows a SQL IDE interface. At the top, there's a 'SQL Commands' tab. Below it, the 'Language' is set to 'SQL' and 'Rows' is set to '10'. The query is as follows:

```
1 SELECT 'The Cost of the' || i.name || 'on this day was' || p.price
2 FROM items i JOIN price_history p
3 ON (TO_DATE('12-Dec-2016', 'DD-MM-YYYY') BETWEEN p.start_date AND p.end_date) AND (i.item_number = 'im01101045')
```

The 'Results' tab is active, showing a single row of data:

'THECOSTOFTHE'    I.NAME    'ONTHISDAYWAS'    P.PRICE
The Cost of theunder shirton this day was14.99

At the bottom, it indicates '1 rows returned in 0.03 seconds' and provides a 'Download' link.

## PART 2:

### Use a Self-Join to Join a Table to Itself

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

The screenshot shows the SQL Developer interface with the following SQL query:

```
1 SELECT r.first_name||' '||r.last_name AS "Sales Representative", s.first_name||' '||s.last_name AS "Supervisor"
2 FROM sales_representatives r JOIN sales_representatives s
3 ON (r.supervisor_id = s.id);
```

The results tab displays the following data:

Sales Representative	Supervisor
Charles Raymond	Charles Raymond
Barry Speed	Charles Raymond
Victoria Wright	Charles Raymond

3 rows returned in 0.02 seconds

### Use OUTER joins

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)

The screenshot shows the SQL Developer interface with the following SQL query:

```
1 SELECT *
2 FROM teams t LEFT OUTER JOIN customers c
3 ON (t.id = c.tem_id)
```

The results tab displays the following data:

ID	NAME	NUMBER_OF_PLAYERS	DISCOUNT	CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID
t001	Rockets	25	10	c00001	bob.thornberry@heatmail.com	Robert	Thornberry	01234567898	150	sr01
t002	Celtics	42	20	c00101	unknown@here.com	John	Doe	03216547808	9875	sr01
t003	Rovers	8	-	c01986	margal87@delphiview.com	Maria	Galant	01442736589	125.65	sr03
t004	Jets	10	5	-	-	-	-	-	-	-

4 rows returned in 0.02 seconds

## Generating a Cartesian Product

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

The screenshot shows the Oracle SQL Developer interface. At the top, the 'SQL Commands' tab is active, displaying the following query:

```
1 SELECT *
2 FROM customers CROSS JOIN sales_representatives;
```

Below the query editor, the 'Results' tab is selected, showing a table with 10 columns: CTR\_NUMBER, EMAIL, FIRST\_NAME, LAST\_NAME, PHONE\_NUMBER, CURRENT\_BALANCE, SRE\_ID, TEM\_ID, LOYALTY\_CARD\_NUMBER, and ID. The table contains 6 rows of data, representing a Cartesian product of the customers and sales\_representatives tables.

CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID	TEM_ID	LOYALTY_CARD_NUMBER	ID
c00012	Jjones@freemail.com	Jennifer	Jones	01505214598	0	-	-	lc1015	sr01
c00001	bob.thornberry@heatmail.com	Robert	Thornberry	01234567898	150	sr01	t001	-	sr01
c02001	brianrog@hooitech.com	Brian	Rogers	01654564898	50	-	-	lc4587	sr01
c00101	unknown@here.com	John	Doe	03216547808	987.5	sr01	t002	-	sr01
c00103	MurciaA@globaltech.com	Andrew	Murcia	07715246890	85	-	-	lc2341	sr01
c01986	marzal87@delohiview.com	Maria	Galant	01442734589	125.65	sr03	t003	-	sr01

The interface also shows the user 'abood053111@gmail.com' and the schema 'WKSP\_DANEY'. The bottom status bar indicates 'Copyright © 1999, 2023, Oracle and/or its affiliates.' and 'Oracle APEX 23.2.1'.