



**UTM**  
UNIVERSITI TEKNOLOGI MALAYSIA

**FACULTY OF COMPUTING**  
UTM Johor Bahru

**SECD2523**  
**SECTION 10**

**LECTURER: ROZILAWATI BINTI DOLLAH @ MD ZAIN**

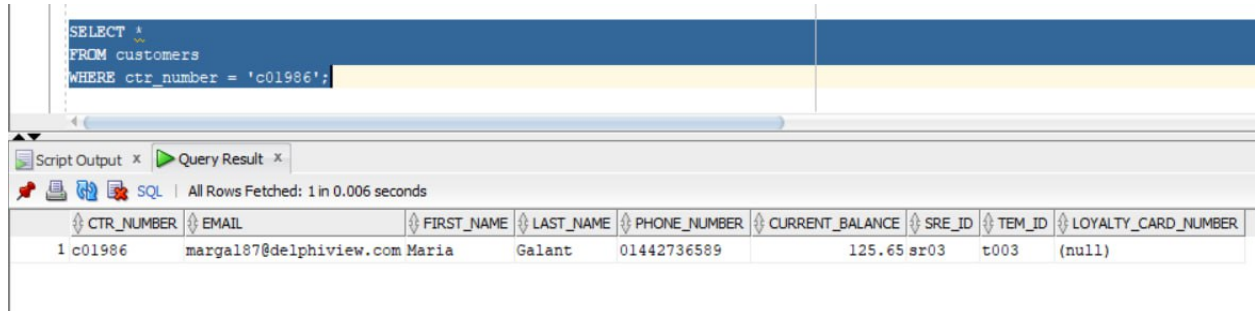
**PROJECT TITLE:**  
**SQL LAB3**  
**(DML2)**  
**PART3**

NAME	MATRIC NUMBER
NADIA SYAHADAH BINTI SAHARUDIN	A22EC0225

## Part 1: Using the WHERE Clause.

1. Using the unique customer number in the where clause display all columns for Maria Galant.

```
SELECT*  
FROM customers  
WHERE ctr_number = 'c01986';
```

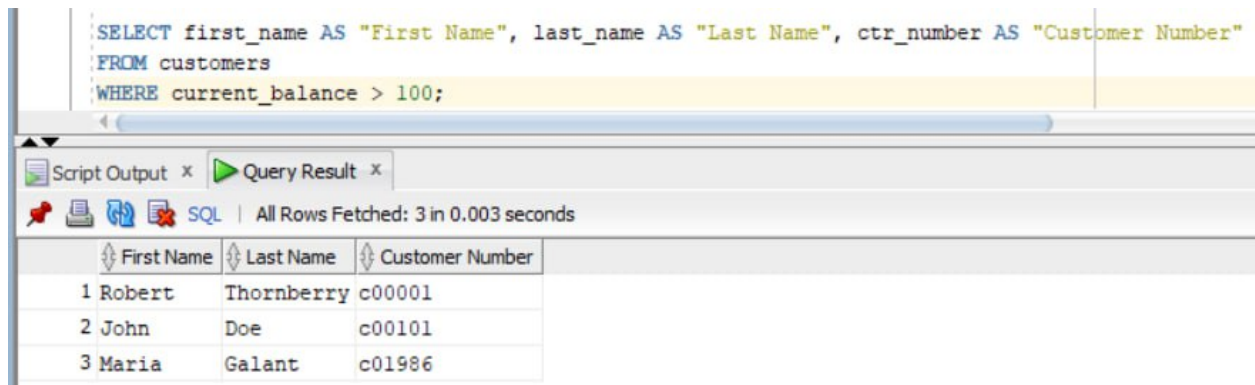


The screenshot shows a SQL query execution window. The query is: `SELECT * FROM customers WHERE ctr_number = 'c01986';`. The result shows 1 row fetched in 0.006 seconds. The table has columns: CTR\_NUMBER, EMAIL, FIRST\_NAME, LAST\_NAME, PHONE\_NUMBER, CURRENT\_BALANCE, SRE\_ID, TEM\_ID, and LOYALTY\_CARD\_NUMBER. The data row is: 1 c01986, margal87@delphiview.com, Maria, Galant, 01442736589, 125.65, sr03, t003, (null).

CTR_NUMBER	EMAIL	FIRST_NAME	LAST_NAME	PHONE_NUMBER	CURRENT_BALANCE	SRE_ID	TEM_ID	LOYALTY_CARD_NUMBER
1 c01986	margal87@delphiview.com	Maria	Galant	01442736589	125.65	sr03	t003	(null)

2. Display the first name, last name and customer number for all customers who have a current balance of greater than 100. Use an appropriate alias for your column headings.

```
SELECT first_name AS "First Name", last_name AS "Last Name",  
ctr_number AS "Customer Number"  
FROM customers  
WHERE current_balance > 100;
```



The screenshot shows a SQL query execution window. The query is: `SELECT first_name AS "First Name", last_name AS "Last Name", ctr_number AS "Customer Number" FROM customers WHERE current_balance > 100;`. The result shows 3 rows fetched in 0.003 seconds. The table has columns: First Name, Last Name, and Customer Number. The data rows are: 1 Robert, Thornberry, c00001; 2 John, Doe, c00101; 3 Maria, Galant, c01986.

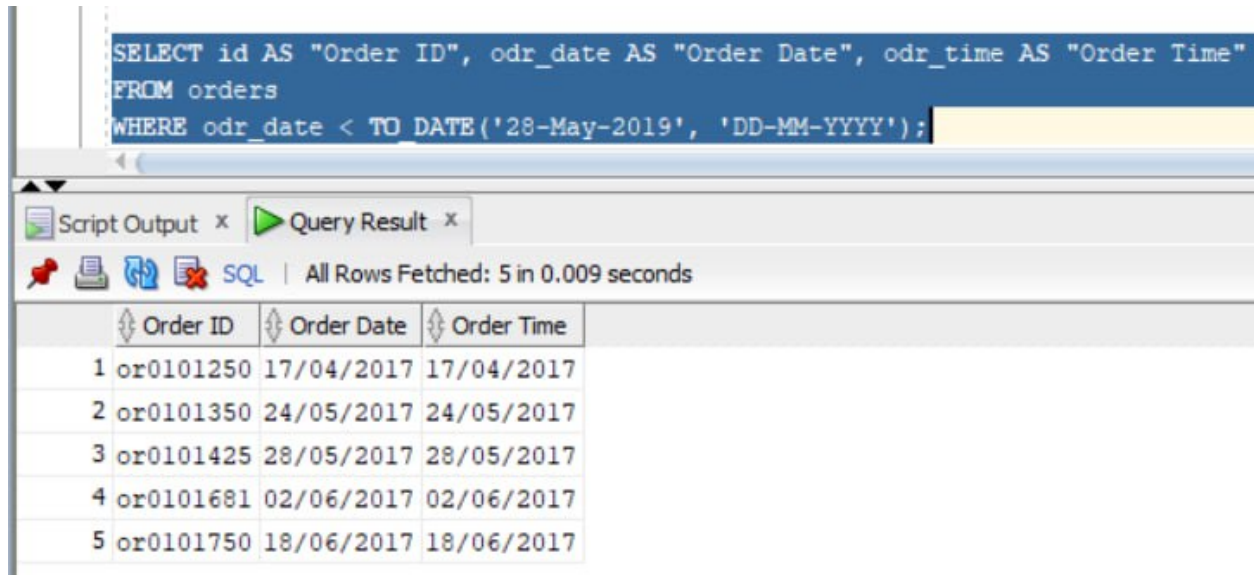
	First Name	Last Name	Customer Number
1	Robert	Thornberry	c00001
2	John	Doe	c00101
3	Maria	Galant	c01986

3. Display the order id, date and time of all orders that were placed before the 28th of May 2019. Use an appropriate alias for your column headings.

```

SELECT id AS "Order ID", odr_date AS "Order Date", odr_time AS
"Order Time"
FROM orders
WHERE odr_date < TO_DATE ('28-05-2019', 'DD-MM-YYYY');

```



The screenshot shows the SQL Developer interface with the query executed. The 'Query Result' tab is active, displaying 5 rows of data. The status bar indicates 'All Rows Fetched: 5 in 0.009 seconds'.

	Order ID	Order Date	Order Time
1	or0101250	17/04/2017	17/04/2017
2	or0101350	24/05/2017	24/05/2017
3	or0101425	28/05/2017	28/05/2017
4	or0101681	02/06/2017	02/06/2017
5	or0101750	18/06/2017	18/06/2017

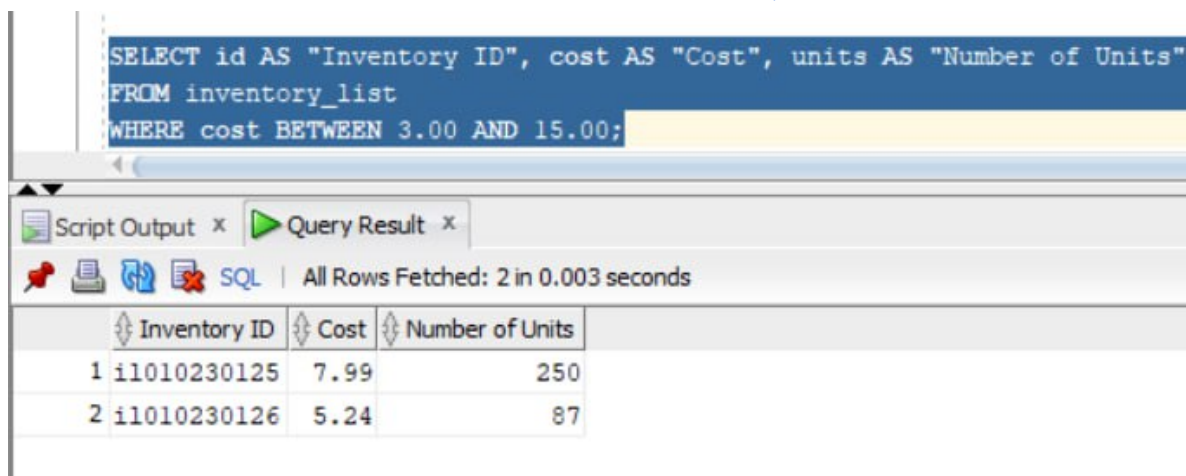
## Part 2: Range Conditions: BETWEEN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that have a trade cost of between 3.00 and 15.00.

```

SELECT id AS "Inventory ID", cost AS "Cost", units AS "Number of
Units"
FROM inventory_list
WHERE cost BETWEEN 3.00 AND 15.00;

```



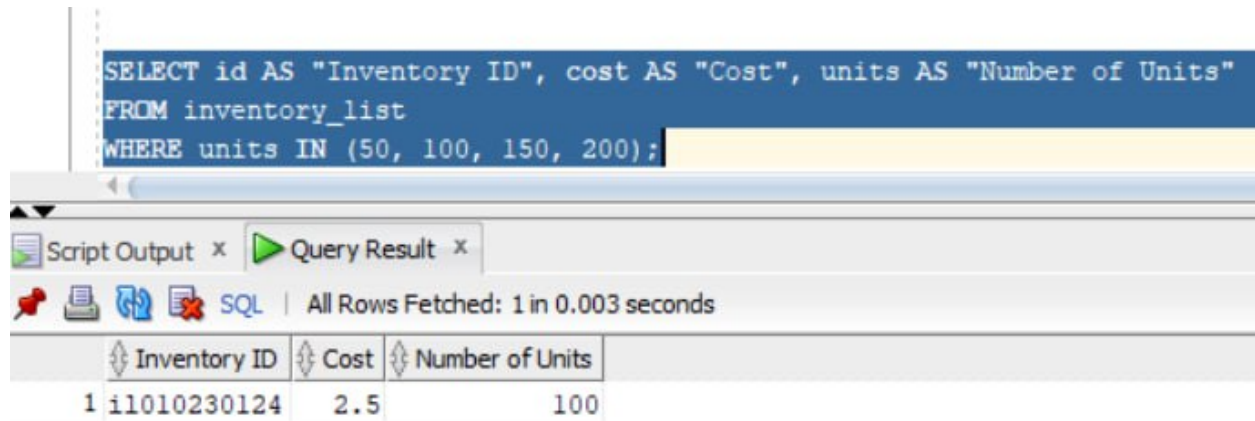
The screenshot shows the SQL Developer interface with the query executed. The 'Query Result' tab is active, displaying 2 rows of data. The status bar indicates 'All Rows Fetched: 2 in 0.003 seconds'.

	Inventory ID	Cost	Number of Units
1	i1010230125	7.99	250
2	i1010230126	5.24	87

### Part 3: Membership Conditions: IN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that have 50, 100, 150 or 200 units in stock.

```
SELECT id AS "Inventory ID", cost AS "Cost", units AS "Number of  
Units"  
FROM inventory_list  
WHERE units IN ('50', '100', '150', '200');
```



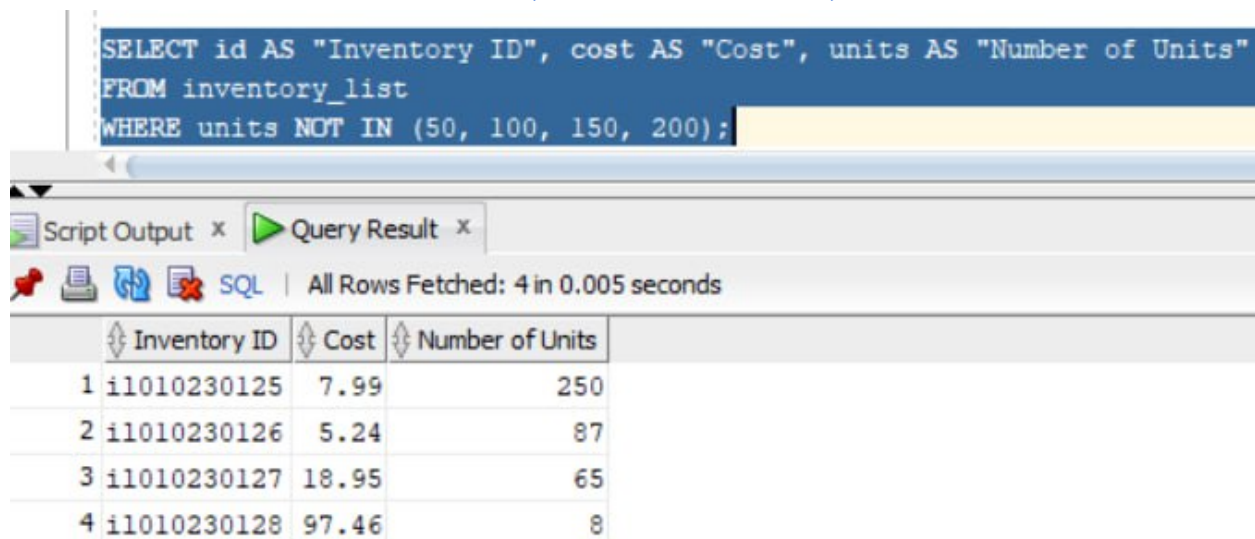
```
SELECT id AS "Inventory ID", cost AS "Cost", units AS "Number of Units"  
FROM inventory_list  
WHERE units IN (50, 100, 150, 200);
```

	Inventory ID	Cost	Number of Units
1	11010230124	2.5	100

### Part 4: Membership Conditions: NOT IN Operator

1. Display the inventory id, cost and number of units using appropriate aliases for all items that do not have 50, 100, 150 or 200 units in stock.

```
SELECT id AS "Inventory ID", cost AS "Cost", units AS "Number of  
Units"  
FROM inventory_list  
WHERE units NOT IN ('50', '100', '150', '200');
```



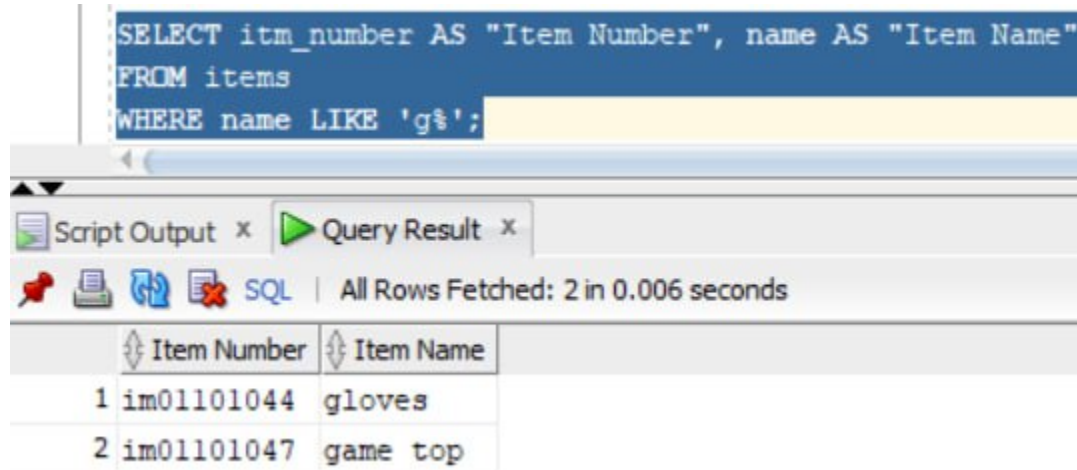
```
SELECT id AS "Inventory ID", cost AS "Cost", units AS "Number of Units"  
FROM inventory_list  
WHERE units NOT IN (50, 100, 150, 200);
```

	Inventory ID	Cost	Number of Units
1	11010230125	7.99	250
2	11010230126	5.24	87
3	11010230127	18.95	65
4	11010230128	97.46	8

## Part 5: Pattern Matching: LIKE Operator

1. Display item number and name of all items that have a name that begins with g. Use an appropriate alias for your column headings.

```
SELECT itm_number AS "Item Number", name AS "Item Name"  
FROM items  
WHERE name LIKE 'g%';
```



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

```
SELECT itm_number AS "Item Number", name AS "Item Name"
FROM items
WHERE name LIKE 'g%';
```

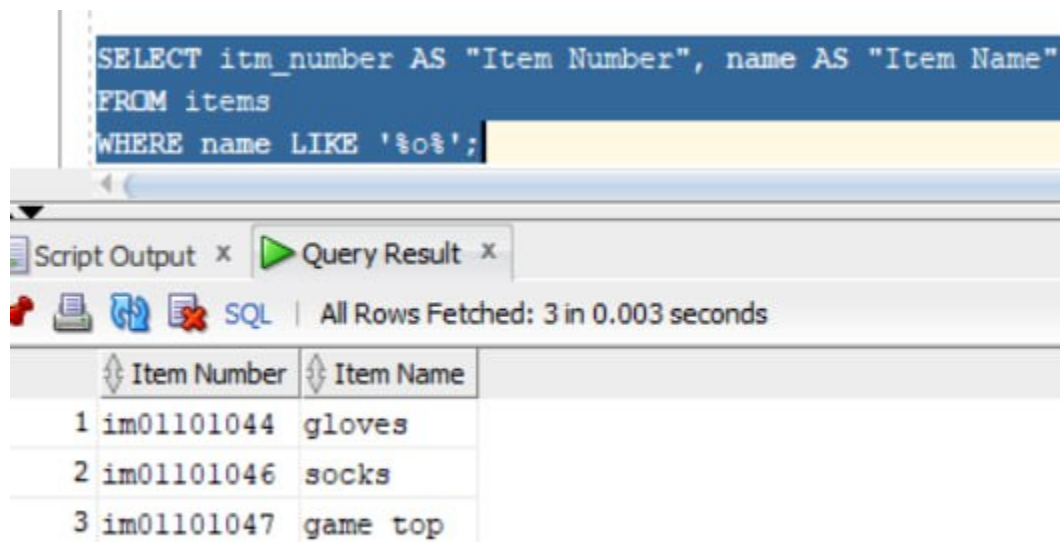
The results window shows the following data:

	Item Number	Item Name
1	im01101044	gloves
2	im01101047	game top

## Part 6 : Pattern Matching: Combining Wildcard Characters with the LIKE Operator

1. Display item number and name of all items that have a name that contain a lowercase o. Use an appropriate alias for your column headings

```
SELECT itm_number AS "Item Number", name AS "Item Name"  
FROM items  
WHERE name LIKE '%o%';
```



The screenshot shows a SQL query editor with a query window and a results window. The query window contains the following SQL code:

```
SELECT itm_number AS "Item Number", name AS "Item Name"
FROM items
WHERE name LIKE '%o%';
```

The results window shows the following data:

	Item Number	Item Name
1	im01101044	gloves
2	im01101046	socks
3	im01101047	game top