

**UJIAN TENGAH SEMESTER  
BASIS DATA**



**DISUSUN OLEH:**

**ABDUL AZIZ  
(244107023009)**

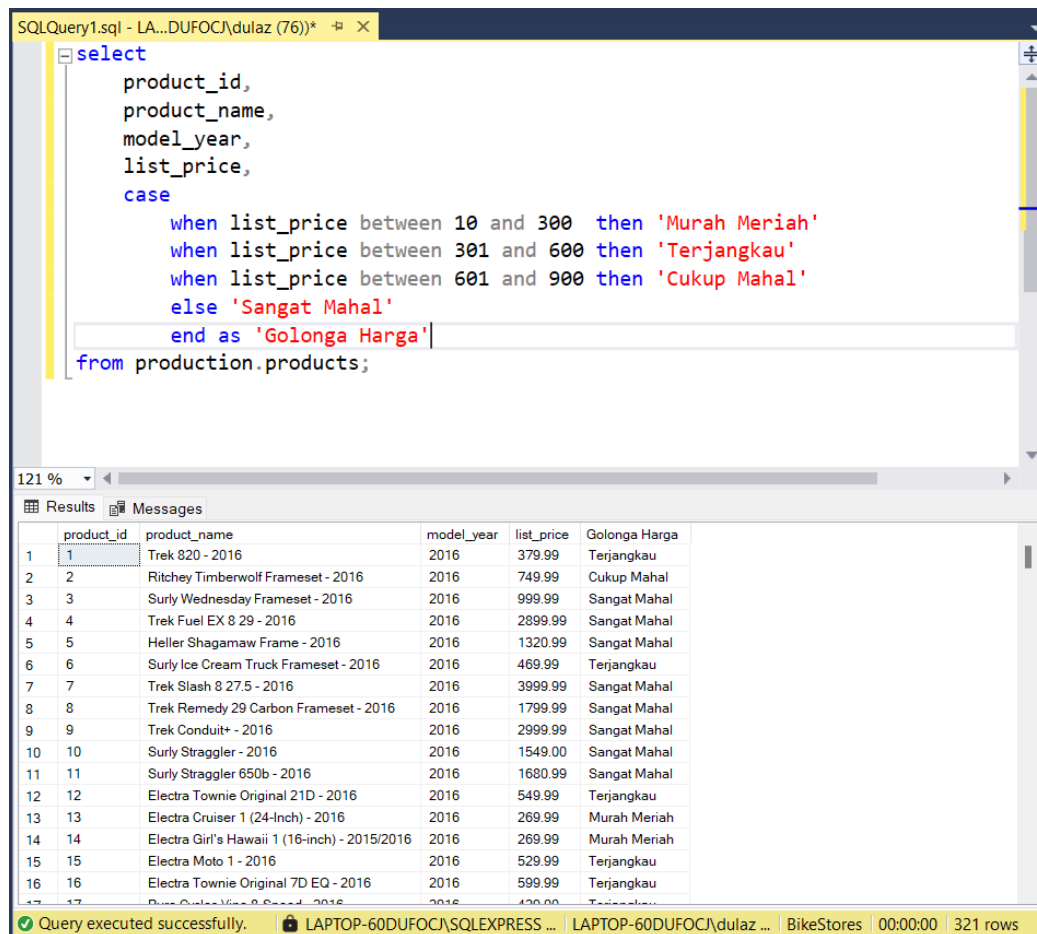
**PROGRAM STUDI D-IV TEKNIK INFORMATIKA  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI MALANG**

## Soal nomor 1

Tampilkan product\_id, product\_name, dan model\_year, list\_price, serta buat alias “Golongan Harga” menggunakan CASE untuk pilihan berikut:

- Jika list\_price diantara 10 sampai 300 maka “Murah meriah”,
- Jika list\_price diantara 301 sampai 600 maka “Terjangkau”,
- Jika list\_price diantara 601 sampai 900 maka “Cukup mahal”,
- Selain itu “Sangat mahal”,

dari tabel production.products. (menghasilkan 321 baris)



The screenshot shows a SQL query window with the following code:

```
select
product_id,
product_name,
model_year,
list_price,
case
when list_price between 10 and 300 then 'Murah Meriah'
when list_price between 301 and 600 then 'Terjangkau'
when list_price between 601 and 900 then 'Cukup Mahal'
else 'Sangat Mahal'
end as 'Golonga Harga'
from production.products;
```

The results pane shows a table with 5 columns: product\_id, product\_name, model\_year, list\_price, and Golonga Harga. The table contains 321 rows of data. The first 16 rows are visible in the screenshot.

product_id	product_name	model_year	list_price	Golonga Harga
1	Trek 820 - 2016	2016	379.99	Terjangkau
2	Ritchey Timberwolf Frameset - 2016	2016	749.99	Cukup Mahal
3	Surly Wednesday Frameset - 2016	2016	999.99	Sangat Mahal
4	Trek Fuel EX 8 29 - 2016	2016	2899.99	Sangat Mahal
5	Heller Shagamaw Frame - 2016	2016	1320.99	Sangat Mahal
6	Surly Ice Cream Truck Frameset - 2016	2016	469.99	Terjangkau
7	Trek Slash 8 27.5 - 2016	2016	3999.99	Sangat Mahal
8	Trek Remedy 29 Carbon Frameset - 2016	2016	1799.99	Sangat Mahal
9	Trek Conduit* - 2016	2016	2999.99	Sangat Mahal
10	Surly Straggler - 2016	2016	1549.00	Sangat Mahal
11	Surly Straggler 650b - 2016	2016	1680.99	Sangat Mahal
12	Electra Townie Original 21D - 2016	2016	549.99	Terjangkau
13	Electra Cruiser 1 (24-inch) - 2016	2016	269.99	Murah Meriah
14	Electra Girl's Hawaii 1 (16-inch) - 2015/2016	2016	269.99	Murah Meriah
15	Electra Moto 1 - 2016	2016	529.99	Terjangkau
16	Electra Townie Original 7D EQ - 2016	2016	599.99	Terjangkau

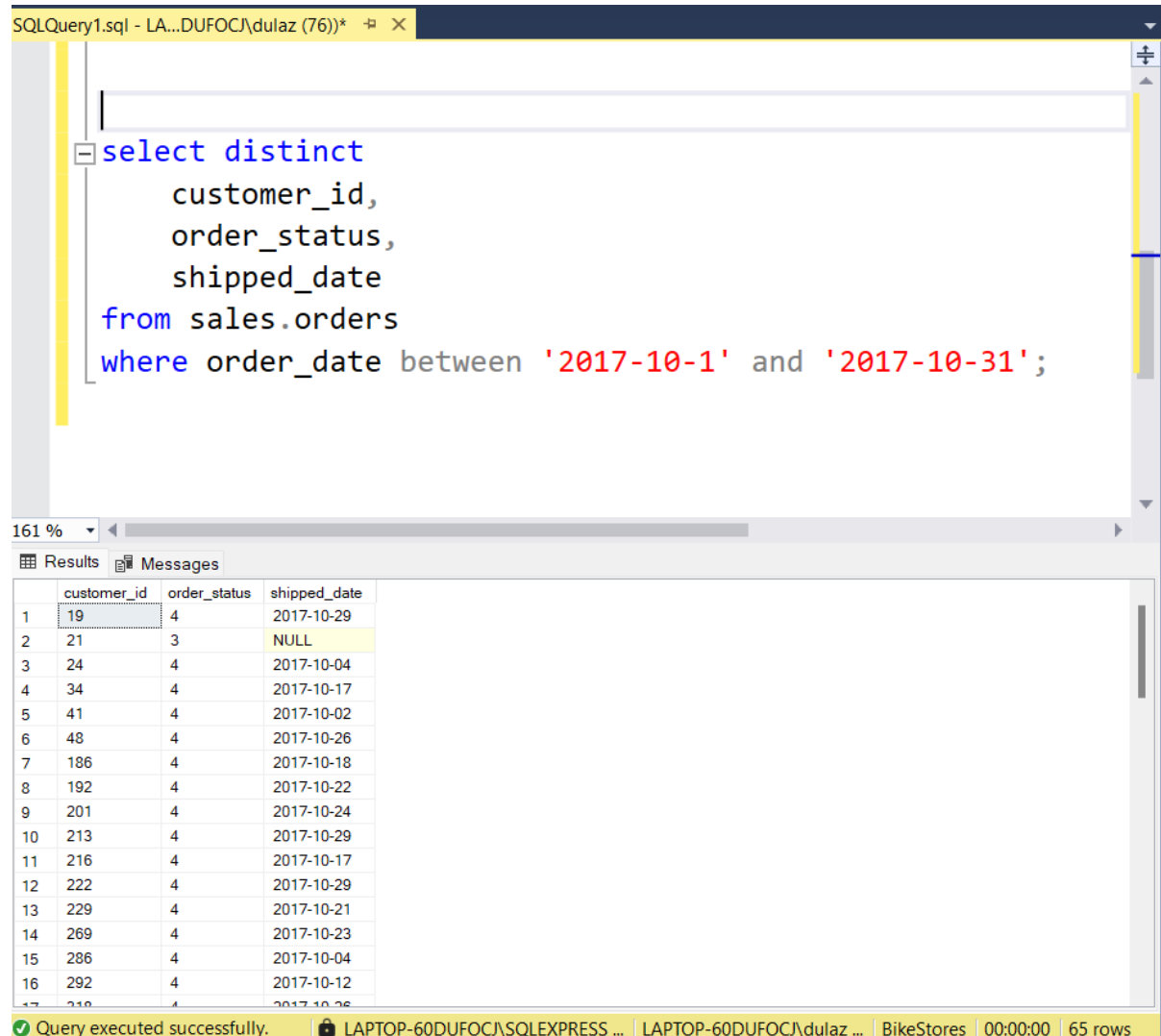
The status bar at the bottom indicates: Query executed successfully. LAPTOP-60DUFOCJ\SQLEXPRESS ... LAPTOP-60DUFOCJ\dulaz ... BikeStores 00:00:00 321 rows

## Penjelasan:

- Mengambil data product\_id, product\_name, model\_year, dan list price dengan cara select kolom – kolom tersebut pada bagian “select”
- Tentukan table yang kita tuju yaitu production.products pada bagian “from”
- Untuk membuat pengkondisiannya bisa menggunakan syntax “case”
- Dengan diawali “case” dan diikuti syntax “when” untuk setiap kondisinya, setelah menentukan kondisi yang diinginkan bisa dilanjutkan dengan syntax “then” untuk melakukan pembuatan output jika kondisi itu terpenuhi
- Jika kondisi tidak ada yang terpenuhi kita bisa membuat opsi dengan menggunakan “else”
- Dan setelah pengkondisian selesai kita bisa menutupnya dengan “end” dan beri nama alias

## Soal nomor 2

Tuliskan query SELECT untuk mendapatkan data unik pada kolom customer\_id, order\_status, shipped\_date dalam tabel sales.orders. Filter hasil tersebut agar hanya menampilkan order pada bulan Oktober 2017 saja. (menghasilkan 66 baris)



The screenshot shows a SQL query window with the following query:

```
select distinct
    customer_id,
    order_status,
    shipped_date
from sales.orders
where order_date between '2017-10-1' and '2017-10-31';
```

The query results are displayed in a table with the following columns: customer\_id, order\_status, and shipped\_date. The results show 65 rows of data.

	customer_id	order_status	shipped_date
1	19	4	2017-10-29
2	21	3	NULL
3	24	4	2017-10-04
4	34	4	2017-10-17
5	41	4	2017-10-02
6	48	4	2017-10-26
7	186	4	2017-10-18
8	192	4	2017-10-22
9	201	4	2017-10-24
10	213	4	2017-10-29
11	216	4	2017-10-17
12	222	4	2017-10-29
13	229	4	2017-10-21
14	269	4	2017-10-23
15	286	4	2017-10-04
16	292	4	2017-10-12
17	310	4	2017-10-28

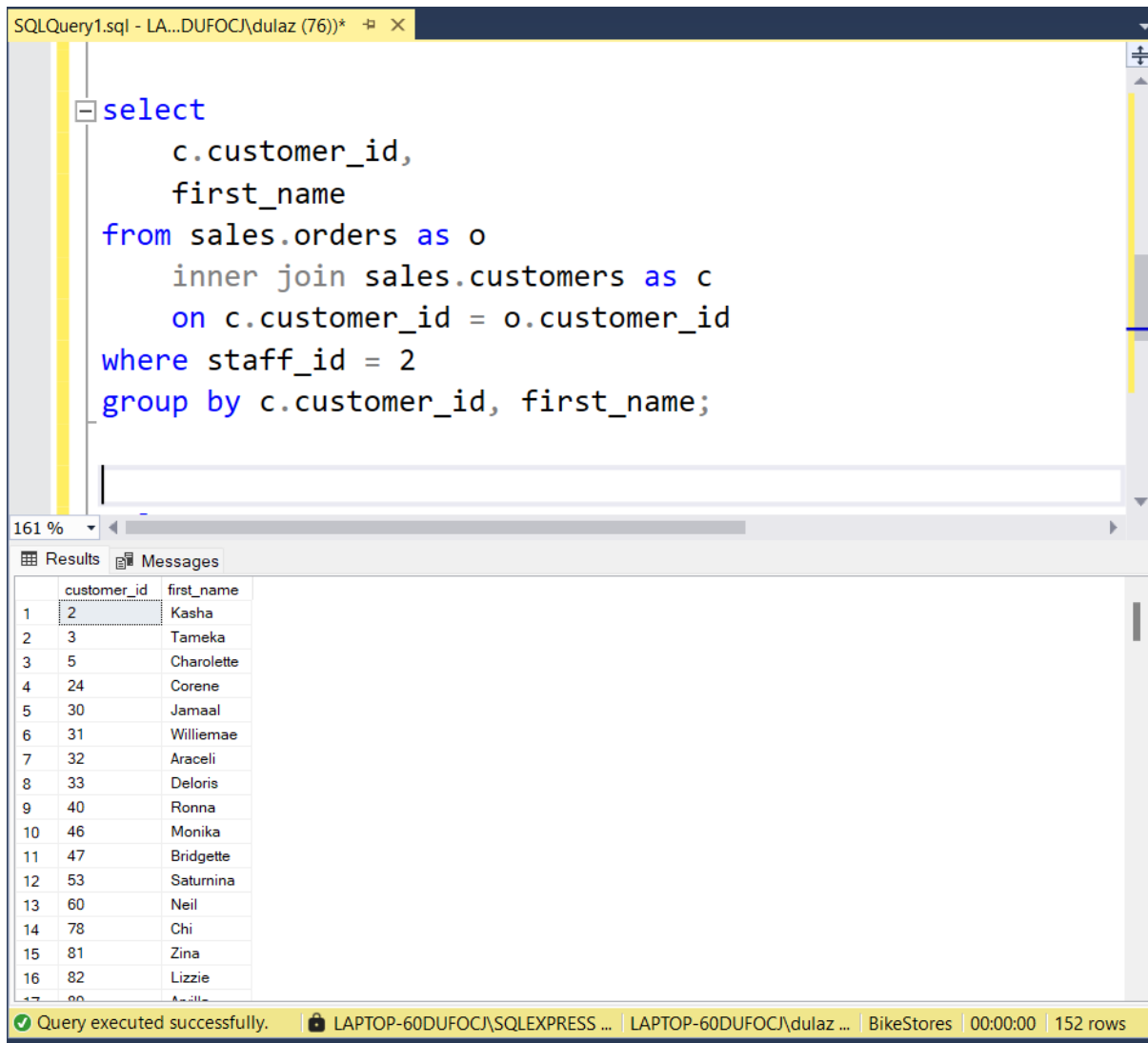
The status bar at the bottom indicates: Query executed successfully. | LAPTOP-60DUFOCJ\SQLEXPRESS ... | LAPTOP-60DUFOCJ\dulaz ... | BikeStores | 00:00:00 | 65 rows

## Penjelasan :

- Dalam menentukan suatu data yang unik kita bisa memakai “distinct”
- Untuk menentukan range waktu disini menggunakan syntax “between” dan diikuti dengan waktu yang pertama kemudian ditambah “and” dan dilanjutkan dengan waktu yang kedua

### Soal nomor 3

Tuliskan T-SQL SELECT yang akan menampilkan kelompok customer yang melakukan order. Klausa SELECT harus mencakup kolom customer\_id dari tabel sales.orders dan kolom first\_name dari tabel sales.customers. Gabungkan kedua kolom tersebut menggunakan INNER JOIN, dan filter hanya order dari staff yang memiliki staff\_id sama dengan 2. (menghasilkan 152 baris)



The screenshot shows a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor contains the following T-SQL code:

```
select
    c.customer_id,
    first_name
from sales.orders as o
    inner join sales.customers as c
    on c.customer_id = o.customer_id
where staff_id = 2
group by c.customer_id, first_name;
```

The results pane displays a table with two columns: customer\_id and first\_name. The table contains 16 rows of data, with the first row highlighted. The status bar at the bottom indicates that the query was executed successfully and returned 152 rows.

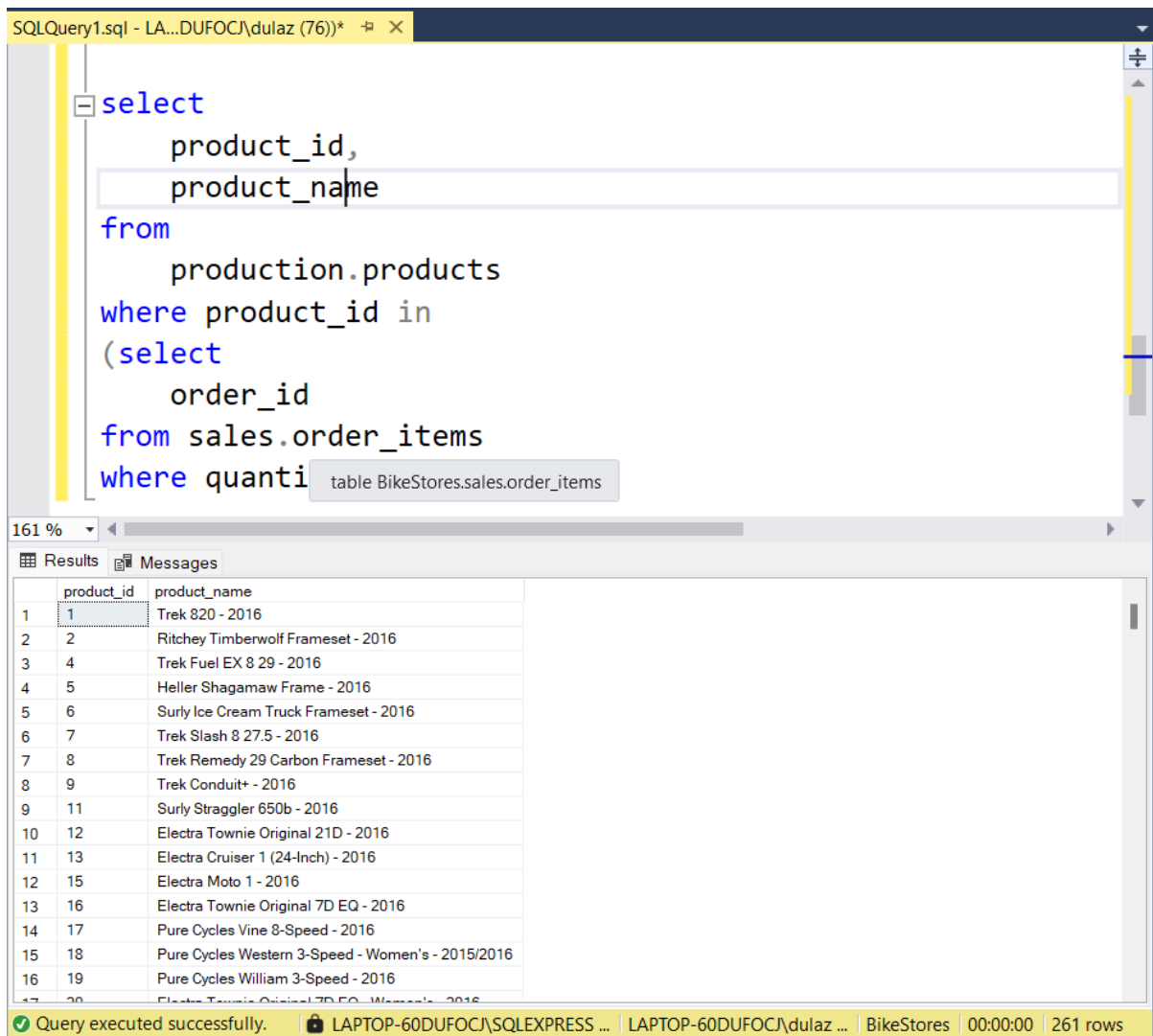
	customer_id	first_name
1	2	Kasha
2	3	Tameka
3	5	Charolette
4	24	Corene
5	30	Jamaal
6	31	Williamae
7	32	Araceli
8	33	Deloris
9	40	Ronna
10	46	Monika
11	47	Bridgette
12	53	Saturnina
13	60	Neil
14	78	Chi
15	81	Zina
16	82	Lizzie

### Penjelasan :

- Karena yang akan didapatkan yaitu sebuah kelompok data maka kita akan menggunakan group by dalam mengelompokkannya
- Jadi dalam menampilkan kolom customer\_id, dan first\_name kita akan melakukan inner join antara table sales.customers dan table sales.orders
- Dalam pemfilteran disini kita menggunakan klausa where dengan staff\_id yaitu 2 dan untuk group by berdasarkan customer\_id dan first\_name

#### Soal nomor 4

Buatlah sub query yang hasilnya untuk menampilkan kolom order\_id dari produk yang terjual dalam jumlah lebih dari 1 kali dari table sales.order\_items. Kemudian buat outer query-nya berdasarkan hasil tersebut untuk mengambil kolom product\_id dan product\_name dari tabel production.products. (menghasilkan 261 baris)



The screenshot shows a SQL query window with the following query:

```
select
    product_id,
    product_name
from
    production.products
where product_id in
(select
    order_id
from sales.order_items
where quanti
```

The query is executed successfully, and the results are displayed in the Results pane. The results show a list of products with their product\_id and product\_name.

product_id	product_name
1	Trek 820 - 2016
2	Ritchey Timberwolf Frameset - 2016
3	Trek Fuel EX 8 29 - 2016
4	Heller Shagamaw Frame - 2016
5	Surly Ice Cream Truck Frameset - 2016
6	Trek Slash 8 27.5 - 2016
7	Trek Remedy 29 Carbon Frameset - 2016
8	Trek Conduit+ - 2016
9	Surly Straggler 650b - 2016
10	Electra Townie Original 21D - 2016
11	Electra Cruiser 1 (24-Inch) - 2016
12	Electra Moto 1 - 2016
13	Electra Townie Original 7D EQ - 2016
14	Pure Cycles Vine 8-Speed - 2016
15	Pure Cycles Western 3-Speed - Women's - 2015/2016
16	Pure Cycles William 3-Speed - 2016
17	Electra Townie Original 7D EQ - Women's - 2016

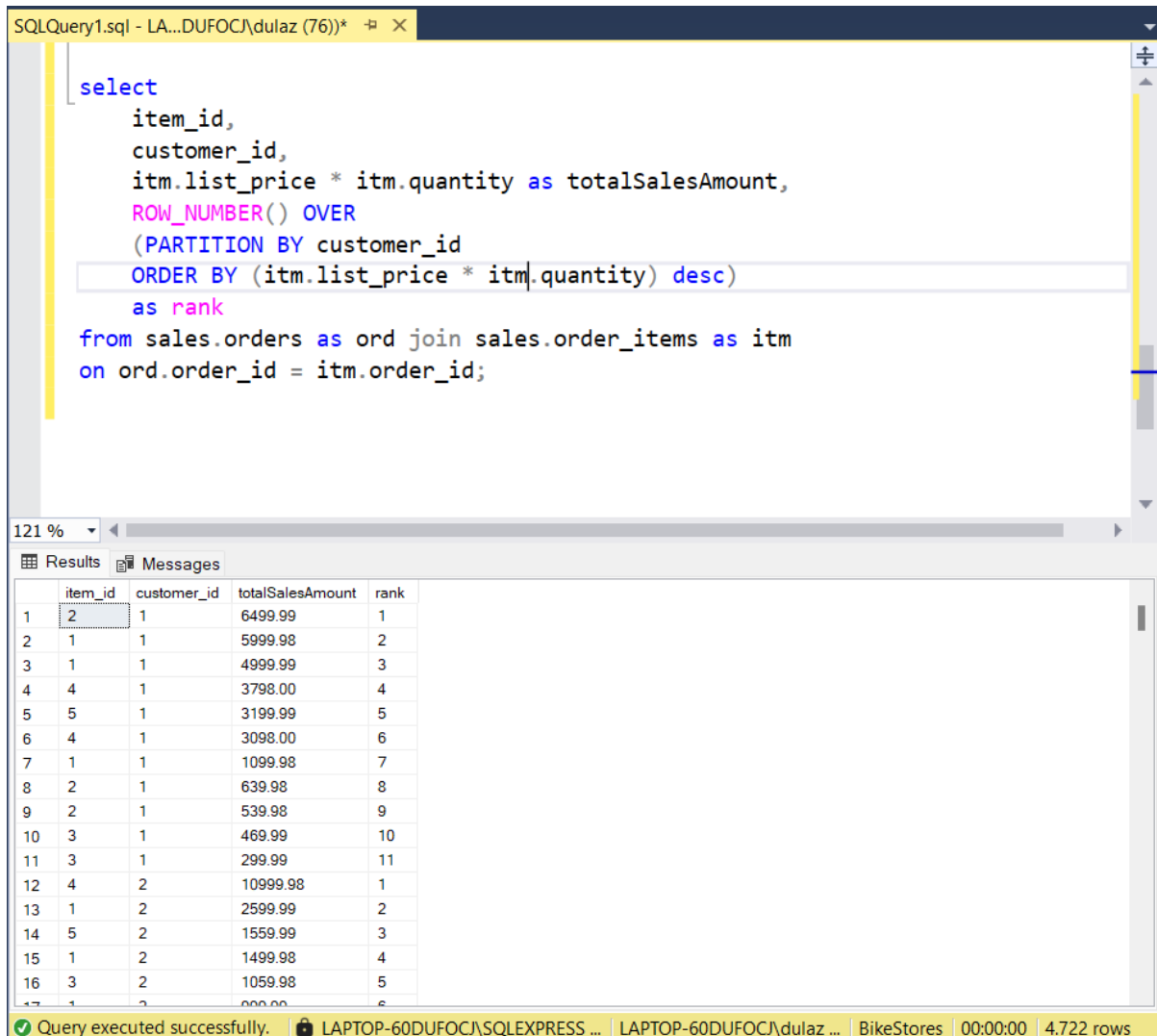
The status bar at the bottom indicates: Query executed successfully. LAPTOP-60DUFOC\SQLEXPRESS ... LAPTOP-60DUFOC\dulaz ... BikeStores | 00:00:00 | 261 rows

#### Penjelasan :

- Kita harus membuat query inner nya terlebih dahulu untuk mendapatkan data order\_id
- Setelah kita mendapatkan order id kita masukan query inner kedalam clause where production\_id dan ditambah syntax "in" yang menandakan bahwa data berdasarkan dari inner query

### Soal nomor 5

Tampilkan kolom `item_id`, `customer_id`, `totalSalesAmount` (`list_price * quantity`), dan perangkainan. Buatlah perangkainan menggunakan `ROW_NUMBER` untuk hasil `list_price` kali `quantity` dari table `sales.order_items`, dipartisi berdasarkan `customer_id`, berdasarkan hasil JOIN table `sales.orders` dengan `sales.order_items`. Diharapkan hasilnya seperti berikut:



The screenshot shows a SQL query window with the following query:

```
select
    item_id,
    customer_id,
    itm.list_price * itm.quantity as totalSalesAmount,
    ROW_NUMBER() OVER
    (PARTITION BY customer_id
    ORDER BY (itm.list_price * itm.quantity) desc)
    as rank
from sales.orders as ord join sales.order_items as itm
on ord.order_id = itm.order_id;
```

The query results are displayed in a table with the following columns: `item_id`, `customer_id`, `totalSalesAmount`, and `rank`. The results are sorted by `rank` in ascending order.

	item_id	customer_id	totalSalesAmount	rank
1	2	1	6499.99	1
2	1	1	5999.98	2
3	1	1	4999.99	3
4	4	1	3798.00	4
5	5	1	3199.99	5
6	4	1	3098.00	6
7	1	1	1099.98	7
8	2	1	639.98	8
9	2	1	539.98	9
10	3	1	469.99	10
11	3	1	299.99	11
12	4	2	10999.98	1
13	1	2	2599.99	2
14	5	2	1559.99	3
15	1	2	1499.98	4
16	3	2	1059.98	5
17	1	2	699.99	6

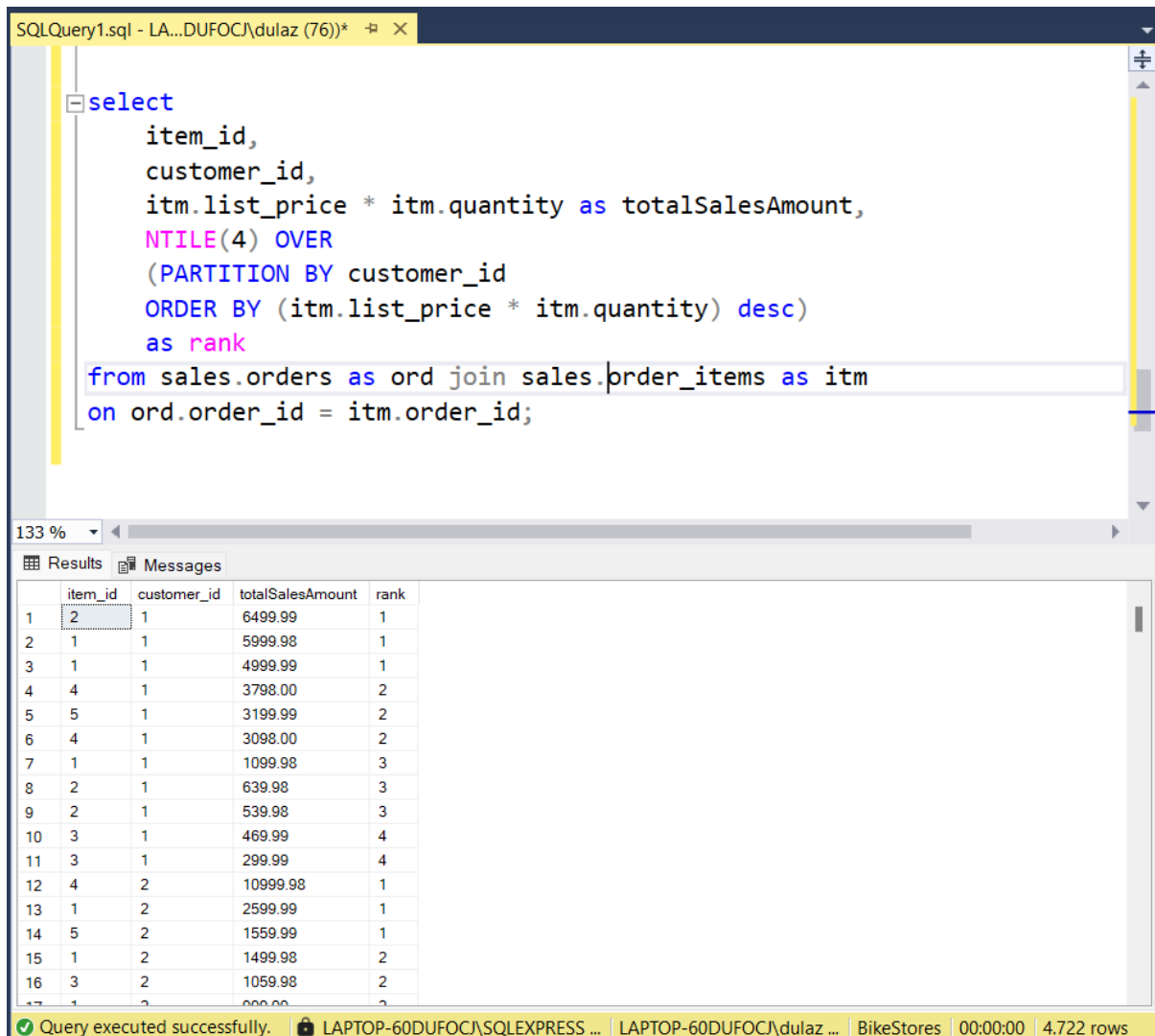
The status bar at the bottom indicates: "Query executed successfully. LAPTOP-60DUFOCJ\SQLEXPRESS ... LAPTOP-60DUFOCJ\dulaz ... BikeStores 00:00:00 4.722 rows".

### Penjelasan :

Implementasi untuk function `ROW_NUMBER()` pada Database BikeStore dengan mengambil data dari 2 table yaitu `sales.orders` dan `sales.order_items` dengan melakukan join pada kedua table tersebut. Data yang ditampilkan yaitu `item_id`, `customer_id`, dan melakukan operasional untuk mendapatkan `totalSalesAmount` yang digunakan untuk mendapatkan ranking dengan partition berdasarkan `customer_id` dengan format DESCENDING. Untuk outputnya akan menampilkan `totalSalesAmount` berserta dengan rankingnya dan tidak adanya duplikasi ranking.

## Soal nomor 6

Berdasarkan soal nomor 5 buatlah perangkian menggunakan NTILE dalam 4 kelompok peringkat. Pastikan hasilnya seperti berikut ini:



The screenshot shows a SQL query window with the following query:

```
select
    item_id,
    customer_id,
    itm.list_price * itm.quantity as totalSalesAmount,
    NTILE(4) OVER
    (PARTITION BY customer_id
    ORDER BY (itm.list_price * itm.quantity) desc)
    as rank
from sales.orders as ord join sales.order_items as itm
on ord.order_id = itm.order_id;
```

The query results are displayed in a table with the following columns: item\_id, customer\_id, totalSalesAmount, and rank. The results are sorted by rank and then by totalSalesAmount.

	item_id	customer_id	totalSalesAmount	rank
1	2	1	6499.99	1
2	1	1	5999.98	1
3	1	1	4999.99	1
4	4	1	3798.00	2
5	5	1	3199.99	2
6	4	1	3098.00	2
7	1	1	1099.98	3
8	2	1	639.98	3
9	2	1	539.98	3
10	3	1	469.99	4
11	3	1	299.99	4
12	4	2	10999.98	1
13	1	2	2599.99	1
14	5	2	1559.99	1
15	1	2	1499.98	2
16	3	2	1059.98	2
17	1	2	999.99	2

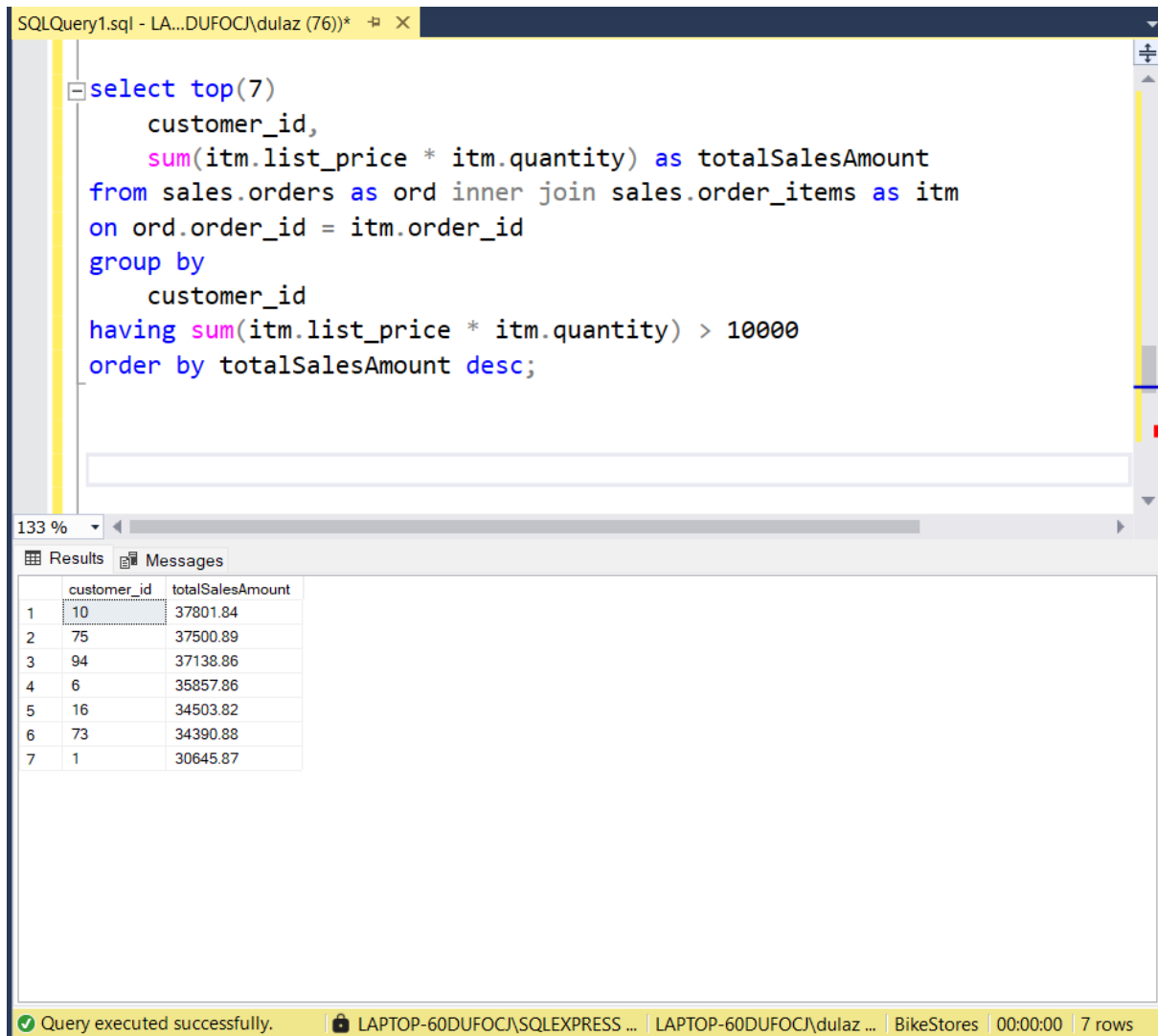
The status bar at the bottom indicates: Query executed successfully. | LAPTOP-60DUFOCJ\SQLEXPRESS ... | LAPTOP-60DUFOCJ\dulaz ... | BikeStores | 00:00:00 | 4,722 rows

### Penjelasan :

Function NTILE() memiliki cara kerja yang berbeda dalam penentuan ranking, fungsi ini akan menentukan berapa jumlah ranking yang akan kita buat dan dalam satu partisi memungkinkan adanya ranking duplikat sesuai dengan data partisi dan ranking yang ditentukan function ini lebih acak dalam menentukan ranking

### Soal nomor 7

Tuliskan perintah T-SQL dengan klausa SELECT untuk mengambil 7 customer teratas dengan penjualan total lebih dari \$10.000. Tampilkan kolom customer\_id dari tabel sales.order dan hitung kolom yang berisi jumlah penjualan berdasarkan kolom quantity dan list\_price dari table sales.order\_items. Gunakan alias totalsalesamount. Diharapkan hasilnya seperti berikut ini



The screenshot shows a SQL Server Enterprise Manager window with a T-SQL query editor and a results pane. The query is as follows:

```
select top(7)
    customer_id,
    sum(itm.list_price * itm.quantity) as totalSalesAmount
from sales.orders as ord inner join sales.order_items as itm
on ord.order_id = itm.order_id
group by
    customer_id
having sum(itm.list_price * itm.quantity) > 10000
order by totalSalesAmount desc;
```

The results pane displays the following data:

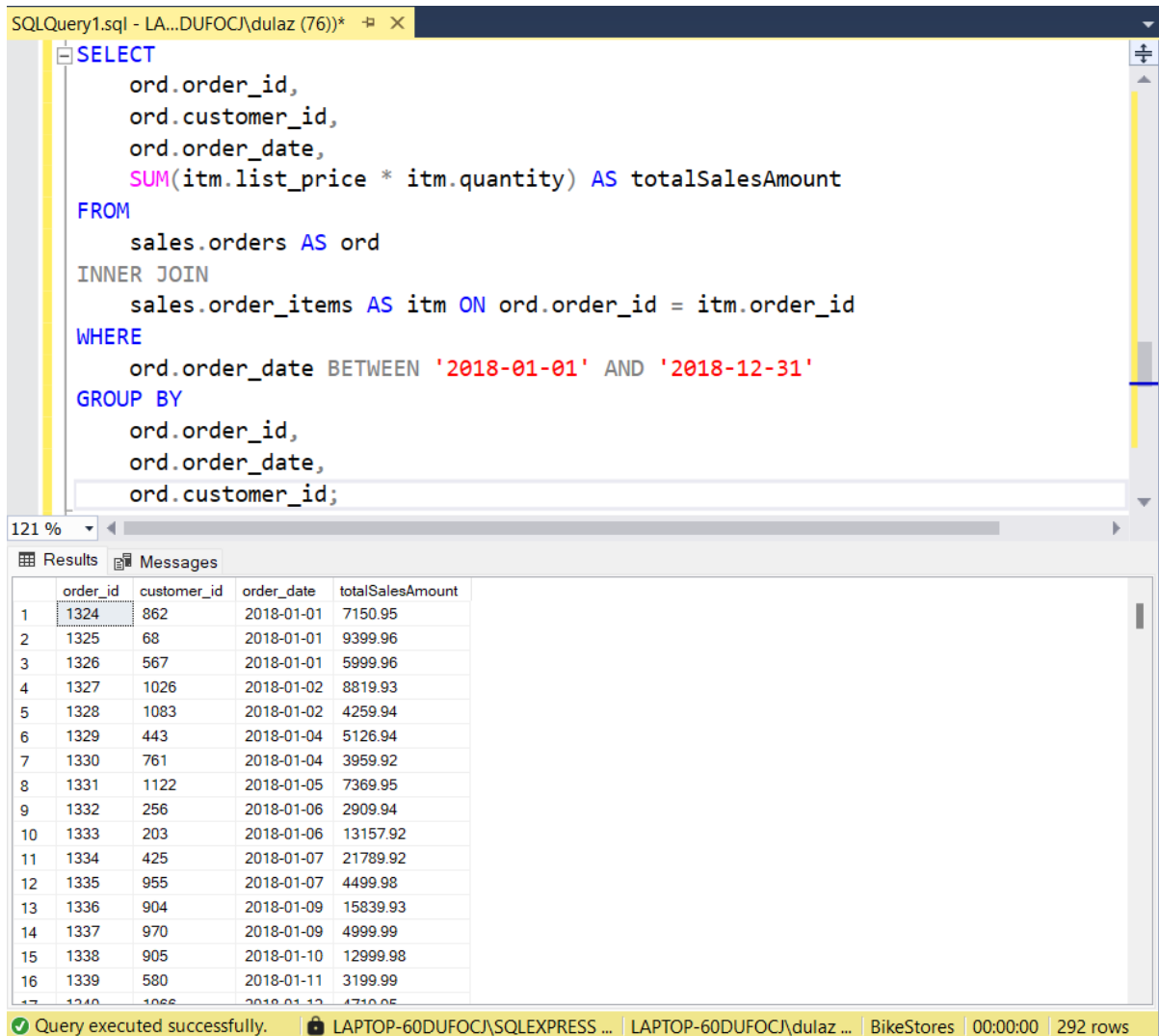
	customer_id	totalSalesAmount
1	10	37801.84
2	75	37500.89
3	94	37138.86
4	6	35857.86
5	16	34503.82
6	73	34390.88
7	1	30645.87

The status bar at the bottom indicates: Query executed successfully. | LAPTOP-60DUFOCJ\SQLEXPRESS ... | LAPTOP-60DUFOCJ\dulaz ... | BikeStores | 00:00:00 | 7 rows



### Soal nomor 8

Tuliskan perintah T-SQL dengan klausa SELECT untuk mengambil kolom order\_id, customer\_id dan kolom yang mempresentasikan perhitungan total penjualan (total sales amount) berdasarkan tabel sales.orders dan sales.order\_items. Filter hasilnya menjadi grup baris data hanya untuk pesanan di tahun 2018! Diharapkan hasilnya seperti berikut ini



The screenshot displays a SQL Server Enterprise Manager window with a query editor and a results pane. The query editor shows a T-SQL query that selects order\_id, customer\_id, order\_date, and the sum of list\_price multiplied by quantity (totalSalesAmount) from sales.orders and sales.order\_items, filtered for the year 2018 and grouped by order\_id, order\_date, and customer\_id.

```
SELECT
    ord.order_id,
    ord.customer_id,
    ord.order_date,
    SUM(itm.list_price * itm.quantity) AS totalSalesAmount
FROM
    sales.orders AS ord
INNER JOIN
    sales.order_items AS itm ON ord.order_id = itm.order_id
WHERE
    ord.order_date BETWEEN '2018-01-01' AND '2018-12-31'
GROUP BY
    ord.order_id,
    ord.order_date,
    ord.customer_id;
```

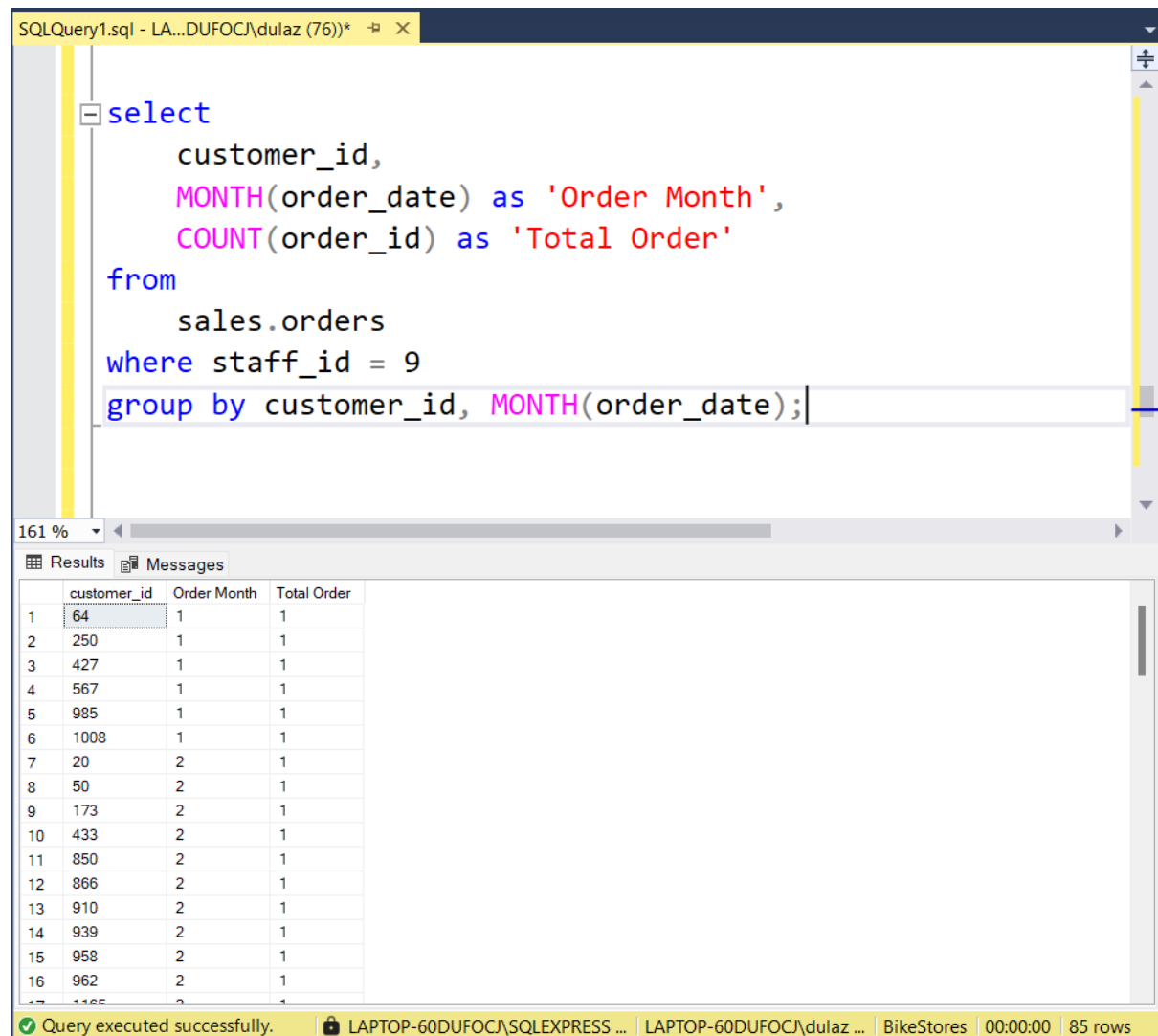
The results pane shows a table with 17 rows and 4 columns: order\_id, customer\_id, order\_date, and totalSalesAmount. The data is grouped by order\_id, order\_date, and customer\_id, showing the total sales amount for each group.

	order_id	customer_id	order_date	totalSalesAmount
1	1324	862	2018-01-01	7150.95
2	1325	68	2018-01-01	9399.96
3	1326	567	2018-01-01	5999.96
4	1327	1026	2018-01-02	8819.93
5	1328	1083	2018-01-02	4259.94
6	1329	443	2018-01-04	5126.94
7	1330	761	2018-01-04	3959.92
8	1331	1122	2018-01-05	7369.95
9	1332	256	2018-01-06	2909.94
10	1333	203	2018-01-06	13157.92
11	1334	425	2018-01-07	21789.92
12	1335	955	2018-01-07	4499.98
13	1336	904	2018-01-09	15839.93
14	1337	970	2018-01-09	4999.99
15	1338	905	2018-01-10	12999.98
16	1339	580	2018-01-11	3199.99
17	1340	1066	2018-01-12	4710.95

Query executed successfully. LAPTOP-60DUFOCJ\SQLEXPRESS ... LAPTOP-60DUFOCJ\dulaz ... BikeStores 00:00:00 292 rows

### Soal nomor 9

Tuliskan pernyataan SELECT yang akan menampilkan kelompok baris berdasarkan kolom customer\_id dan akan dihitung oleh kolom ordermonth mewakili bulan order berdasarkan kolom order\_date dari tabel sales.orders. Kemudian filter hasilnya untuk memasukkan hanya order dari staff yang sama dengan 9! (menghasilkan 85 baris)



The screenshot shows a SQL query window with the following query:

```
select
    customer_id,
    MONTH(order_date) as 'Order Month',
    COUNT(order_id) as 'Total Order'
from
    sales.orders
where staff_id = 9
group by customer_id, MONTH(order_date);
```

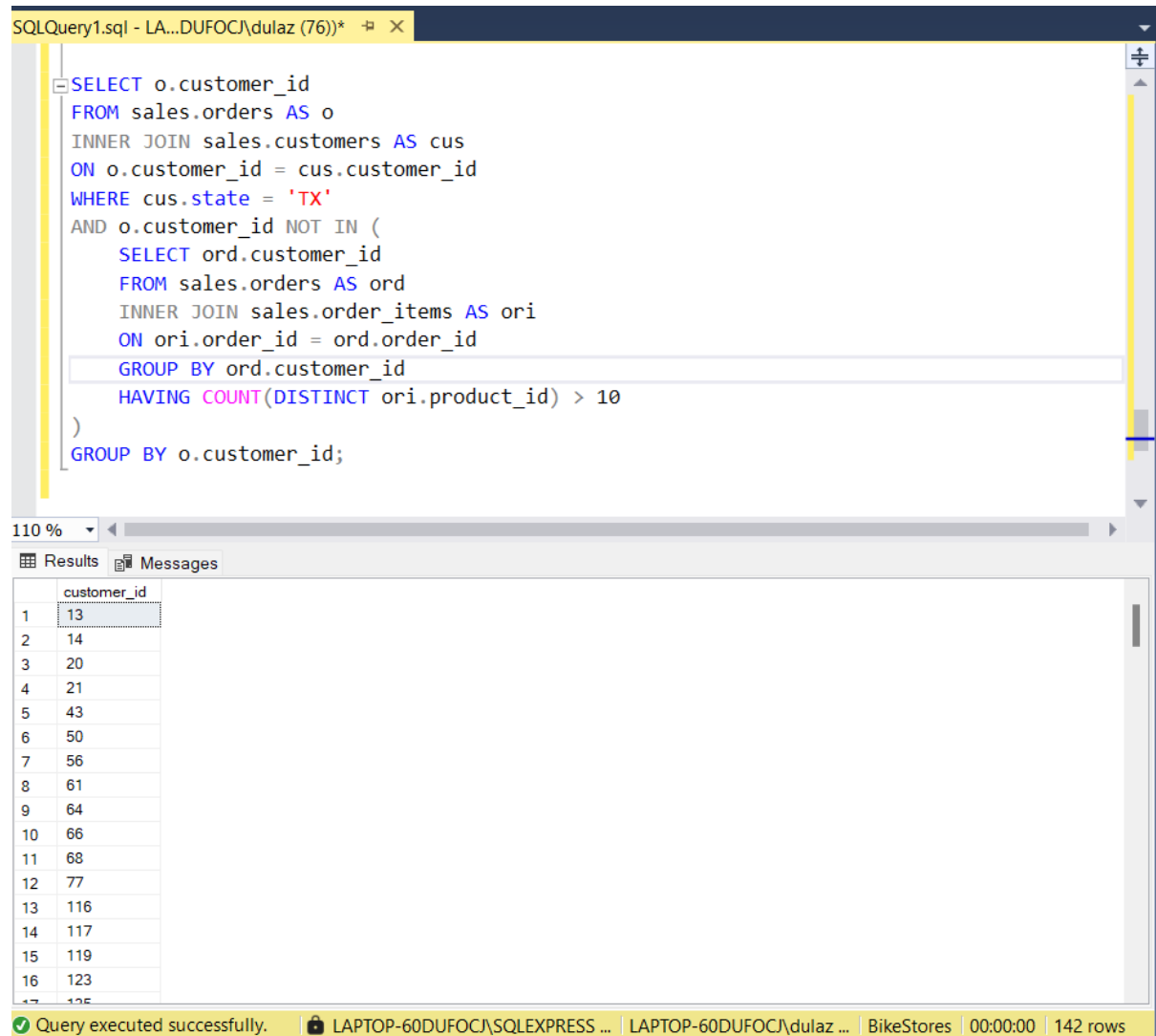
The query is executed successfully, and the results are displayed in a table with 85 rows. The table has four columns: customer\_id, Order Month, and Total Order. The results show that for each customer\_id, there is one row for each month from 1 to 12, with a total order count of 1 for each month.

	customer_id	Order Month	Total Order
1	64	1	1
2	250	1	1
3	427	1	1
4	567	1	1
5	985	1	1
6	1008	1	1
7	20	2	1
8	50	2	1
9	173	2	1
10	433	2	1
11	850	2	1
12	866	2	1
13	910	2	1
14	939	2	1
15	958	2	1
16	962	2	1
17	1165	2	1

Query executed successfully. LAPTOP-60DUFOCJ\SQLEXPRESS ... LAPTOP-60DUFOCJ\dulaz ... BikeStores 00:00:00 85 rows

### Soal nomor 10

Buatlah sebuah statement SELECT yang menampilkan kolom 'customer\_id' dari tabel 'sales.orders'. Saring hasilnya sehingga yang tampil hanyalah pelanggan yang berasal dari state = TX kecuali SEMUA pelanggan yang muncul pada hasil query diatas. (menghasilkan 142 baris)



The screenshot shows a SQL query window with the following query:

```
SELECT o.customer_id
FROM sales.orders AS o
INNER JOIN sales.customers AS cus
ON o.customer_id = cus.customer_id
WHERE cus.state = 'TX'
AND o.customer_id NOT IN (
    SELECT ord.customer_id
    FROM sales.orders AS ord
    INNER JOIN sales.order_items AS ori
    ON ori.order_id = ord.order_id
    GROUP BY ord.customer_id
    HAVING COUNT(DISTINCT ori.product_id) > 10
)
GROUP BY o.customer_id;
```

The query is executed successfully, and the results are displayed in a table with 142 rows. The table has a single column named 'customer\_id'.

	customer_id
1	13
2	14
3	20
4	21
5	43
6	50
7	56
8	61
9	64
10	66
11	68
12	77
13	116
14	117
15	119
16	123
17	125

The status bar at the bottom indicates: Query executed successfully. LAPTOP-60DUFOCJ\SQLEXPRESS ... LAPTOP-60DUFOCJ\dulaz ... BikeStores | 00:00:00 | 142 rows