

**Nama : Ibnu Rusdianto**

**Kelas : Full Stack 1**

**Tugas : Sesi 20 - Kamis, 05 Okt 2023 – MySQL**

**Jenis Tugas : Individu**

**Link Youtube (Penjelasan Query) : [I'am just a poor programmer](#)**

**Link Youtube (Penjelasan SQL Join) : [I'am just a poor programmer](#)**

**Download file .sql Apotek : [Apotek.sql](#)**



### 1. Query Insert

```
INSERT INTO obat (id_obat, nama_obat, expired_obat, jenis_obat, dosis_obat)
VALUES (1, 'Bioderma Sensibio', '2024-09-10', 'membersihkan wajah', '2');

INSERT INTO obat (id_obat, nama_obat, expired_obat, jenis_obat, dosis_obat)
VALUES (1, 'Bodrex Strip 10 Tablet', '2024-09-10', 'sakit kepala', '1 atau 2 tablet setiap 4-6 jam');
```

### 2. Query Update

```
apotek> UPDATE obat SET dosis_obat = '2 Setiap 4-7 jam sehari' WHERE id_obat = 1
[2023-09-29 08:26:51] 1 row affected in 3 ms
apotek> UPDATE obat SET nama_obat = 'Bodrex Strip' WHERE id_obat = 2
[2023-09-29 08:26:51] 1 row affected in 5 ms

UPDATE obat SET dosis_obat = '2 Setiap 4-7 jam sehari' WHERE id_obat = 1;
UPDATE obat SET nama_obat = 'Bodrex Strip' WHERE id_obat = 2;
```

### 3. Query Delete

```
aptek> DELETE FROM obat WHERE id_obat = 1
[2023-09-29 08:29:52] 1 row affected in 8 ms
```

```
DELETE FROM obat WHERE id_obat = 1;
```

#### 4. SELECT Query

Output apotek.pasien x

4 rows v

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Heru Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City,...	87134887
2	2	Ibnu Rusdianto	L	20	Subang, Pusakajaya, Cigugur Kidul	85134887
3	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa ...	82134867
4	4	Kireina	P	2	Subang, Pusakajaya, Cigugur Kaler	82134887

```
1 ✓ SELECT * FROM pasien
```

```
1 ✓ SELECT nama, alamat FROM pasien
```

	nama	alamat
1	Heru Rusdianto	Jatihandap, Mandalajati, Bandung City, West Java 40195
2	Ibnu Rusdianto	Subang, Pusakajaya, Cigugur Kidul
3	Ravindra El-Barra	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia
4	Kireina	Subang, Pusakajaya, Cigugur Kaler

#### 5. WHERE Query

```
1 ✓ SELECT * FROM pasien
2 WHERE alamat LIKE 'j%';
```

Output apotek.pasien x

1 row v

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Heru Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887

```
1 ✓ SELECT * FROM pasien
2 WHERE umur BETWEEN 10 AND 20;
```

Output apotek.pasien

	id_pasien	nama	sex	umur	alamat
1	2	Ibnu Rusdianto	L	20	Subang, Pusakajaya, Cigugur Kidul

## 6. AND Query

```
1 ✓ SELECT nama, sex FROM pasien WHERE alamat='Subang, Pusakajaya, Cigugur Kaler' AND sex='P';
2
```

Output apotek.pasien

	nama	sex
1	Kireina	P

```
1 ✓ SELECT * FROM pasien
2 WHERE sex = 'L'
3 AND alamat = 'Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia';
4
```

Output apotek.pasien

	id_pasien	nama	sex	umur	alamat	no_hp
1	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia	82134867

## 7. Sintak OR Query

```
1 ✓ SELECT sex,nama FROM pasien WHERE alamat='Subang, Pusakajaya, Cigugur Kidul' OR sex='P';
2
```

Output apotek.pasien x

2 rows

	sex	nama
1	L	Ibnu Rusdianto
2	P	Kireina

console x

```

1 ✓ SELECT nama, alamat, no_hp FROM pasien
2   WHERE alamat = 'Jatihandap, Mandalajati, Bandung City, West Java 40195'
3   AND nama LIKE 'h%' OR nama LIKE 'i%';

```

Output apotek.pasien x

2 rows

	nama	alamat	no_hp
1	Heru Rusdianto	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887
2	Ibnu Rusdianto	Subang, Pusakajaya, Cigugur Kidul	85134887

## 8. WHERE NOT Query

console x

```

1 ✓ SELECT * FROM pasien
2   WHERE NOT alamat = 'Subang, Pusakajaya, Cigugur Kidul';

```

Output apotek.pasien x

3 rows

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Heru Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887
2	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia	82134867
3	4	Kireina	P	2	Subang, Pusakajaya, Cigugur Kaler	82134887

```
console x
Tx: Auto ✓
Playground
1 ✓ SELECT * FROM pasien
2 WHERE NOT umur < 5;
3
```

Output apotek.pasien

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Herv Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887
2	2	Ibnu Rusdianto	L	20	Subang, Pusakajaya, Cigugur Kidul	85134887
3	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia	82134867

## 9. ORDER BY Query

```
console x
Tx: Auto ✓
Playground
1 ✓ SELECT * FROM pasien
2 ORDER BY alamat ASC , nama DESC;
3
```

Output apotek.pasien

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Herv Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887
2	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia	82134867
3	4	Kireina	P	2	Subang, Pusakajaya, Cigugur Kaler	82134887
4	2	Ibnu Rusdianto	L	20	Subang, Pusakajaya, Cigugur Kidul	85134887

```
console x
Tx: Auto ✓
Playground
1 ✓ SELECT * FROM pasien
2 ORDER BY nama, alamat;
3
```

	id_pasien	nama	sex	umur	alamat	no_hp
1	1	Heru Rusdianto	L	25	Jatihandap, Mandalajati, Bandung City, West Java 40195	87134887
2	2	Ibnu Rusdianto	L	20	Subang, Pusakajaya, Cigugur Kidul	85134887
3	4	Kireina	P	2	Subang, Pusakajaya, Cigugur Kaler	82134887
4	3	Ravindra El-Barra	L	5	Kecamatan Coblong, Kota Bandung, Jawa Barat, Indonesia	82134867

## 10. MIN Query

```

console x
[play] [clock] [p] [gear] Tx: Auto ✓ [refresh] [stop] Playground [table]
1 ✓ SELECT MIN(expired_obat) FROM obat;
2

```

	MIN(expired_obat):date
1	2024-09-10

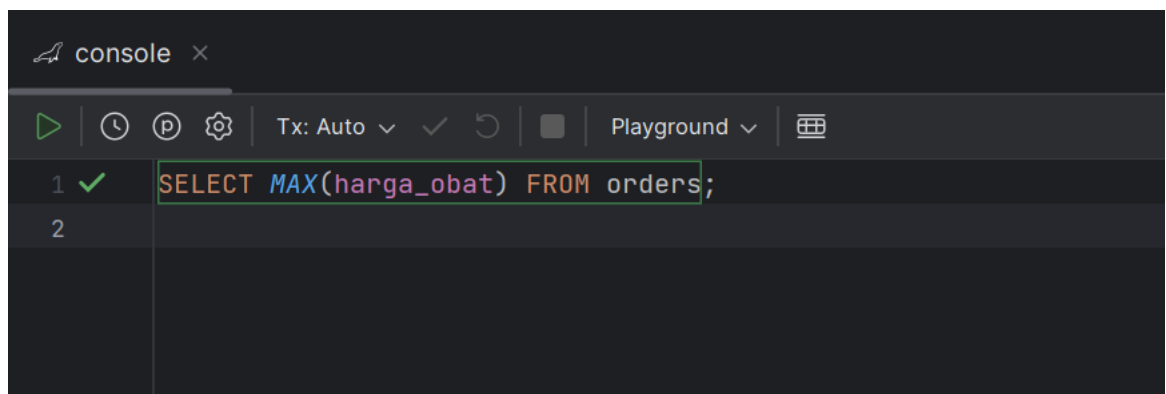
```

console x
[play] [clock] [p] [gear] Tx: Auto ✓ [refresh] [stop] Playground [table]
1 ✓ SELECT MIN(umur) FROM pasien;
2

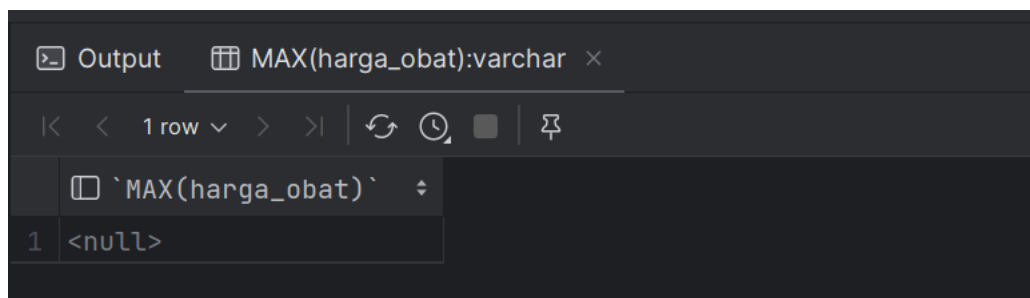
```

	MIN(umur):int(2)
1	2

## 11. MAX Query

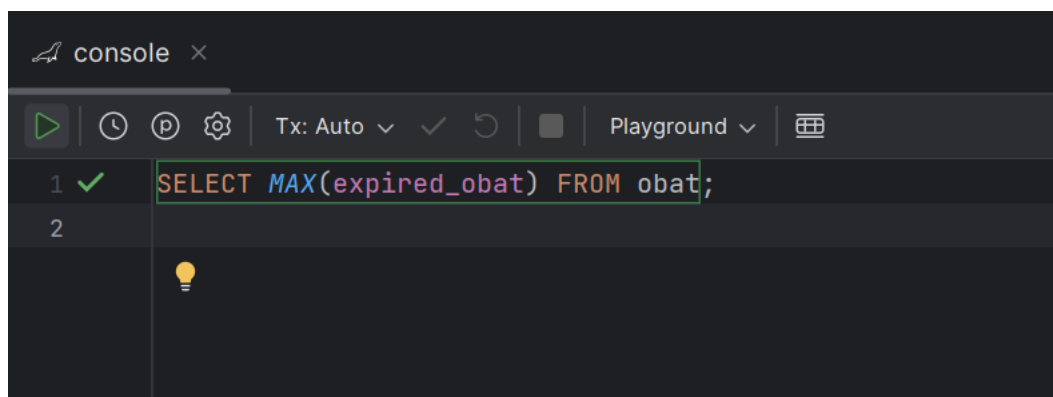


A screenshot of a SQL console interface. The console has a tab labeled 'console' with a close button. Below the tab is a toolbar with icons for running, refreshing, and other functions, along with a dropdown menu set to 'Tx: Auto'. The main area shows a SQL query: `SELECT MAX(harga_obat) FROM orders;`. The query is highlighted with a green border. To the left of the query, there are two line numbers, 1 and 2, with a green checkmark next to line 1.

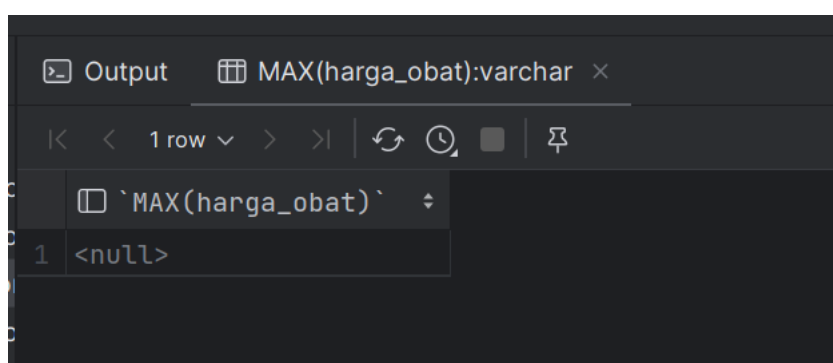


A screenshot of the SQL output window. The window has a tab labeled 'Output' and a dropdown menu showing 'MAX(harga\_obat):varchar'. Below the tab is a toolbar with icons for navigating through the results. The main area shows a table with one row and one column. The column is labeled 'MAX(harga\_obat)' and the row contains the value '<null>'. The table is shown in a compact view.

	MAX(harga_obat)
1	<null>



A screenshot of a SQL console interface. The console has a tab labeled 'console' with a close button. Below the tab is a toolbar with icons for running, refreshing, and other functions, along with a dropdown menu set to 'Tx: Auto'. The main area shows a SQL query: `SELECT MAX(expired_obat) FROM obat;`. The query is highlighted with a green border. To the left of the query, there are two line numbers, 1 and 2, with a green checkmark next to line 1. A lightbulb icon is visible below the query, indicating a suggestion or tip.



A screenshot of the SQL output window. The window has a tab labeled 'Output' and a dropdown menu showing 'MAX(harga\_obat):varchar'. Below the tab is a toolbar with icons for navigating through the results. The main area shows a table with one row and one column. The column is labeled 'MAX(harga\_obat)' and the row contains the value '<null>'. The table is shown in a compact view.

	MAX(harga_obat)
1	<null>

## 12. COUNT Query

console x

Tx: Auto ✓ ↺ | Playground ▾ |

```
1 ✓ SELECT COUNT(*) FROM pasien WHERE sex = 'L';
```

2

Output COUNT(\*):int x

1 row ▾ | |

	`COUNT(*)` ▾
1	3

console x

Tx: Auto ▾ ✓ ↺ | Playground ▾ |

```
1 ✓ SELECT COUNT(*) FROM pasien;
```

2

Output COUNT(\*):int x

1 row ▾ | |

	`COUNT(*)` ▾
1	4

### 13. AVG Query



console x

Tx: Auto ✓ ↺ | Playground ▾ |

```
1 ✓ SELECT AVG(umur) FROM pasien;
```

Output **AVG(umur):decimal** x

1 row ▾ | |

	<b>`AVG(umur)`</b> ▾
1	13.0000

console x

Tx: Auto ✓ ↺ | Playground ▾ |

```
1 ✓ SELECT AVG(umur) FROM pasien WHERE sex = 'L';
```

Output **AVG(umur):decimal** x

1 row ▾ | |

	<b>`AVG(umur)`</b> ▾
1	16.6667

## 14. SUM Query

console x

Tx: Auto ✓ ↺ Playground

```
1 ✓ SELECT SUM(harga_obat) FROM orders WHERE id_pasien = 1;
2
```

💡

Output SUM(harga\_obat):varchar(255) x

1 row

	SUM(harga_obat)
1	<null>

console x

Tx: Auto ✓ ↺ Playground

```
1 ✓ SELECT SUM(qty) FROM order_details;
2
```

💡

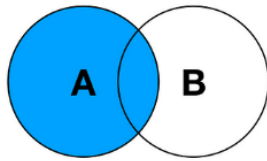
Output SUM(qty):varchar(255) x

1 row

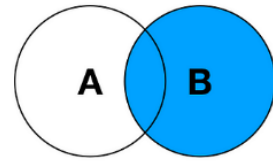
	SUM(qty)
1	<null>

## SQL JOIN QUERY

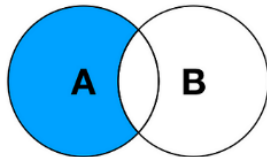
### SQL JOINS



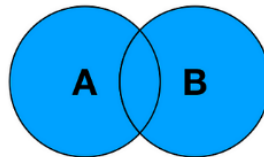
LEFT JOIN



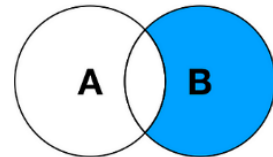
RIGHT JOIN



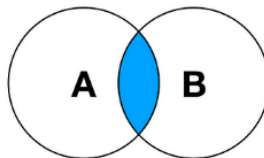
LEFT JOIN EXCLUDING  
INNER JOIN



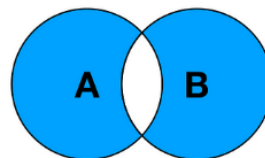
FULL OUTER JOIN



RIGHT JOIN EXCLUDING  
INNER JOIN



INNER JOIN



FULL OUTER JOIN EXCLUDING  
INNER JOIN

15. Inner Join antara orders dan pasien berdasarkan id\_pasien

console x

```
1 ✓ SELECT orders.*, pasien.nama
2 FROM orders
3 INNER JOIN pasien ON orders.id_pasien = pasien.id_pasien;
4
```

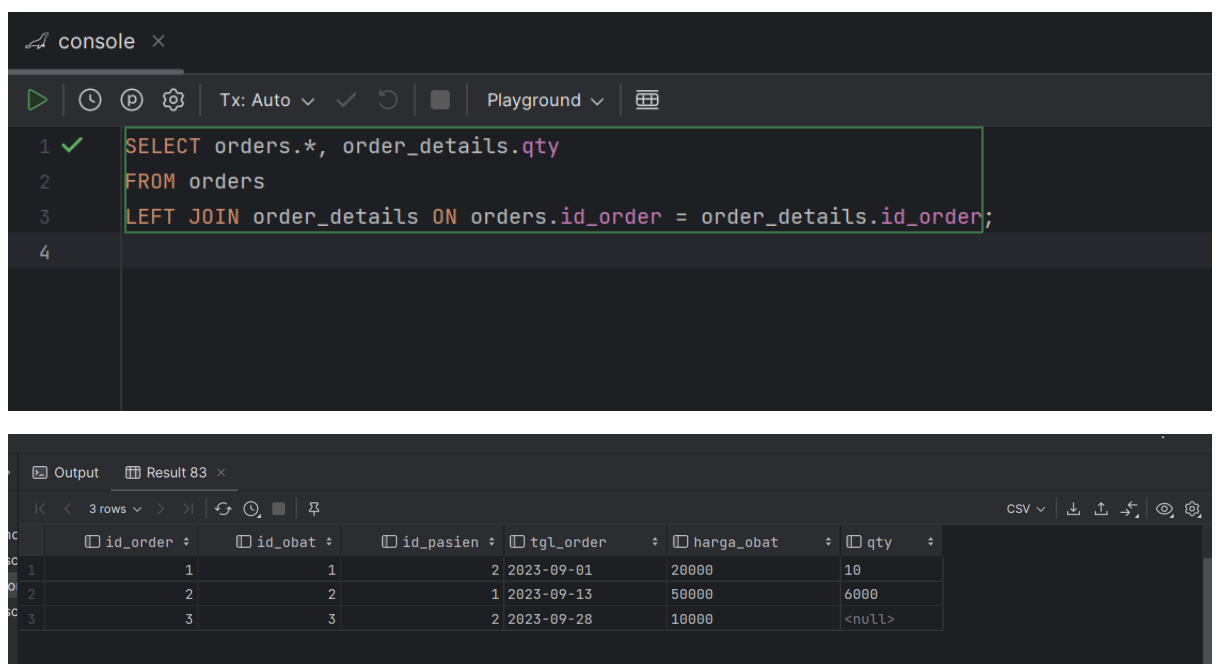
Output Result 81 x

	id_order	id_obat	id_pasien	tgl_order	harga_obat	nama
1	2	2	1	2023-09-13	50000	Heru Rusdianto
2	1	1	2	2023-09-01	20000	Ibnu Rusdianto
3	3	3	2	2023-09-28	10000	Ibnu Rusdianto

Menggabungkan dari 2 table orders dan pasien tetapi berdasarkan `orders.id_pasien = pasien.id_pasien`, maka output dari query tersebut yaitu menampilkan semua kolom dari table orders tetapi ditambahkan nama dari table pasien

"Sistem tolong gabungkan 2 table orders dan pasien tetapi saya ingin berdasarkan `id_pasien` dari orders = `id_pasien` dari pasien"

16. Left Join antara orders dan order\_details berdasarkan `id_order`



```
1 ✓ SELECT orders.*, order_details.qty
2 FROM orders
3 LEFT JOIN order_details ON orders.id_order = order_details.id_order;
4
```

	id_order	id_obat	id_pasien	tgl_order	harga_obat	qty
1	1	1	2	2023-09-01	20000	10
2	2	2	1	2023-09-13	50000	6000
3	3	3	2	2023-09-28	10000	<null>

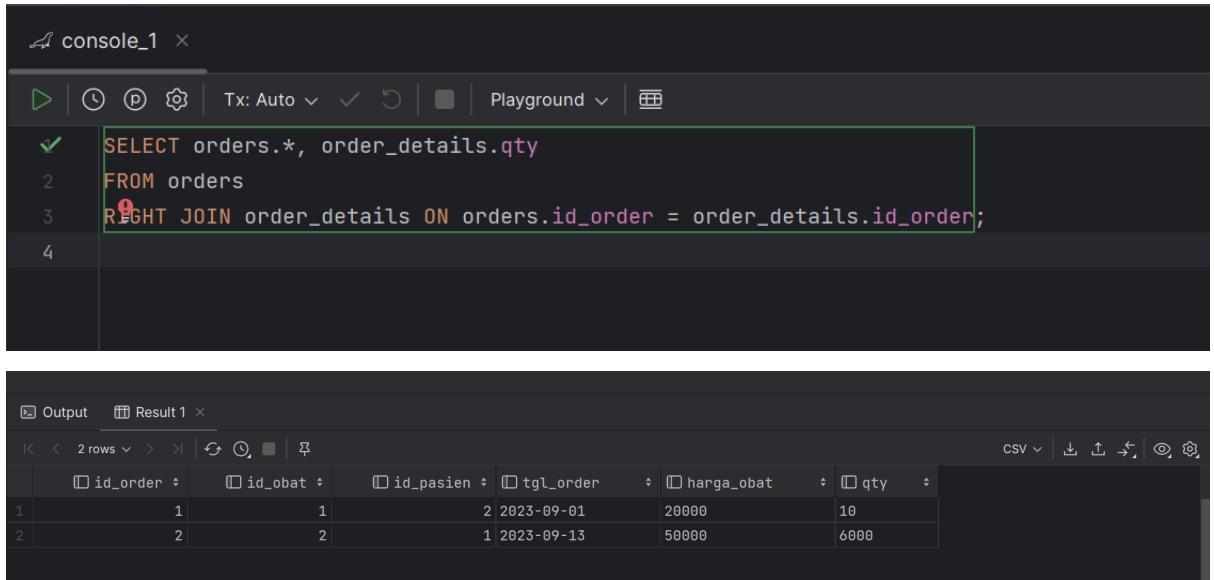
Menggabungkan semua table kiri dari orders tetapi yang sesuai dari tabel sebelah kanan yaitu orders\_details yang berdasarkan `orders.id_orders = order_details.id_order`

hasil dari query diatas adalah menampilkan semua dari table orders tetapi akan menambahkan qty dari table orders\_details.

"Sistem tolong tampilkan semua data pada table orders, dan tampilkan order\_details tetapi hanya qty saja dari table orders, lalu tolong gabungkan table dari ke kiri (orders) tetapi harus sesuai dari sebelah

kanan atau table kanan yaitu order\_details.qty, dimana yang berdasarkan orders.id\_order = order\_details.id\_order"

#### 17. Right Join antara orders dan order\_details berdasarkan id\_order



The screenshot shows a SQL playground interface. The top part displays the SQL query: `SELECT orders.*, order_details.qty FROM orders RIGHT JOIN order_details ON orders.id_order = order_details.id_order;`. The bottom part shows the output table with 2 rows and 6 columns: id\_order, id\_obat, id\_pasien, tgl\_order, harga\_obat, and qty. The first row has values (1, 1, 2, 2023-09-01, 20000, 10) and the second row has values (2, 2, 1, 2023-09-13, 50000, 6000).

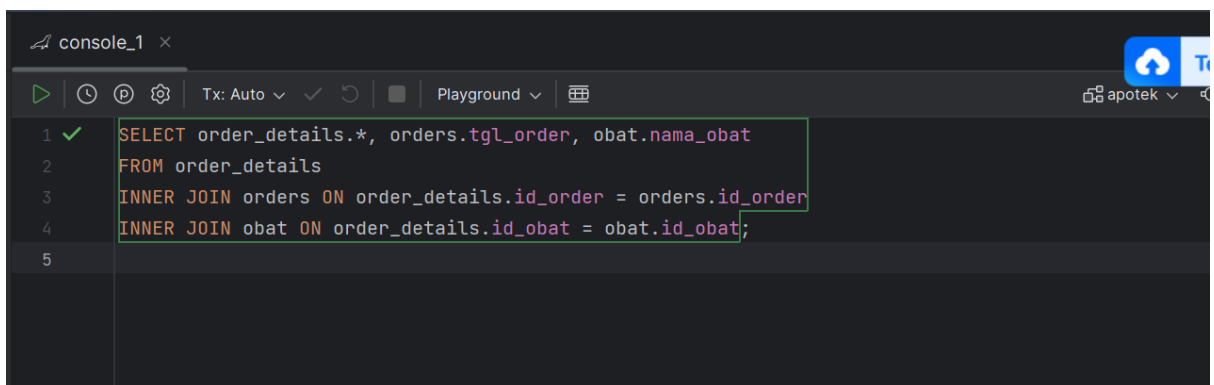
```
SELECT orders.*, order_details.qty
FROM orders
RIGHT JOIN order_details ON orders.id_order = order_details.id_order;
```

	id_order	id_obat	id_pasien	tgl_order	harga_obat	qty
1	1	1	2	2023-09-01	20000	10
2	2	2	1	2023-09-13	50000	6000

Menggabungkan semua table dari sebelah kanan (order\_details) dengan baris yang sesuai dari table sebelah kiri (orders) tetapi berdasarkan orders.id\_order = order\_details.id\_order.

hasil ini akan menampilkan semua kolom pada table orders tetapi ditambah qty dari table order\_details

#### 18. Inner Join antara order\_details, orders, dan obat berdasarkan id\_order dan id\_obat



The screenshot shows a SQL playground interface. The top part displays the SQL query: `SELECT order_details.*, orders.tgl_order, obat.nama_obat FROM order_details INNER JOIN orders ON order_details.id_order = orders.id_order INNER JOIN obat ON order_details.id_obat = obat.id_obat;`. The bottom part shows the output table with 2 rows and 6 columns: id\_order, id\_obat, id\_pasien, tgl\_order, harga\_obat, and qty. The first row has values (1, 1, 2, 2023-09-01, 20000, 10) and the second row has values (2, 2, 1, 2023-09-13, 50000, 6000).

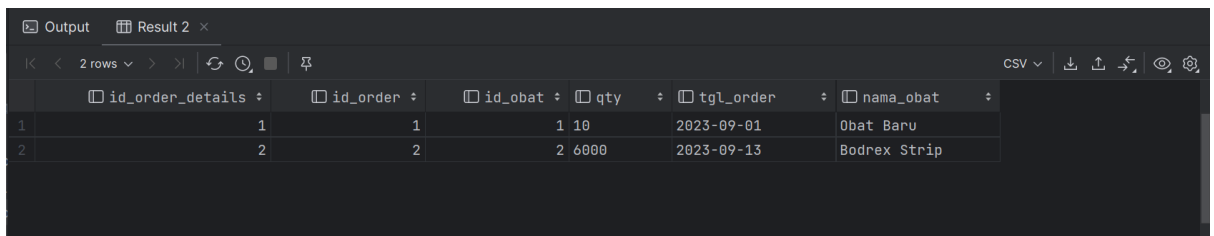
```
SELECT order_details.*, orders.tgl_order, obat.nama_obat
FROM order_details
INNER JOIN orders ON order_details.id_order = orders.id_order
INNER JOIN obat ON order_details.id_obat = obat.id_obat;
```

Menggabungkan Inner Join 3 table yaitu order\_details, orders dan obat, dengan sebuah kondisi yaitu order\_details.id\_order = orders.id\_order, dan order\_details.id\_obat = obat.id\_obat

kondisi pertama : order\_details.id\_order = orders.id\_order

kondisi kedua : order\_details.id\_obat = obat.id\_obat

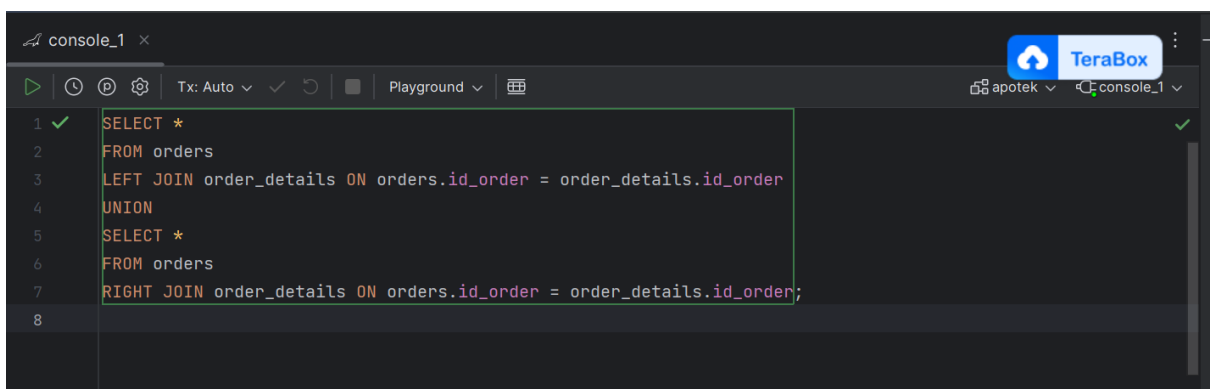
hasil query ini akan menampilkan semua kolom dari table order\_details, tgl\_order dari table orders dan nama\_obat dari table obat



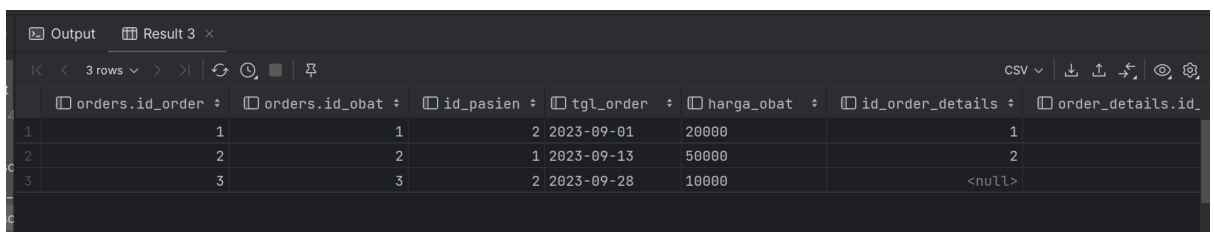
The screenshot shows a database query result with 2 rows. The columns are id\_order\_details, id\_order, id\_obat, qty, tgl\_order, and nama\_obat. The first row shows order\_details.id\_order = 1, id\_order = 1, id\_obat = 1, qty = 10, tgl\_order = 2023-09-01, and nama\_obat = Obat Baru. The second row shows order\_details.id\_order = 2, id\_order = 2, id\_obat = 2, qty = 6000, tgl\_order = 2023-09-13, and nama\_obat = Bodrex Strip.

	id_order_details	id_order	id_obat	qty	tgl_order	nama_obat
1	1	1	1	10	2023-09-01	Obat Baru
2	2	2	2	6000	2023-09-13	Bodrex Strip

## 19. Menggabungkan LEFT & RIGTH JOIN Dengan Metode UNION



```
SELECT *
FROM orders
LEFT JOIN order_details ON orders.id_order = order_details.id_order
UNION
SELECT *
FROM orders
RIGHT JOIN order_details ON orders.id_order = order_details.id_order;
```



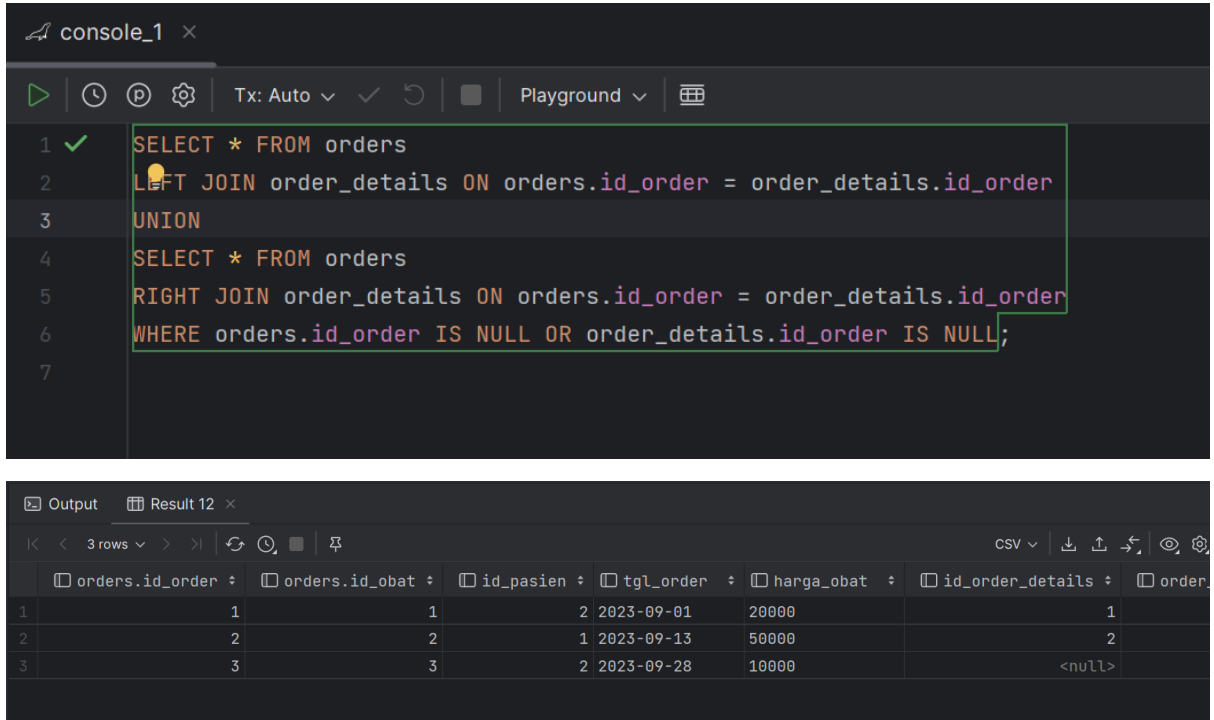
The screenshot shows a database query result with 3 rows. The columns are orders.id\_order, orders.id\_obat, id\_pasien, tgl\_order, harga\_obat, id\_order\_details, and order\_details.id. The first row shows orders.id\_order = 1, orders.id\_obat = 1, id\_pasien = 2, tgl\_order = 2023-09-01, harga\_obat = 20000, id\_order\_details = 1, and order\_details.id = 1. The second row shows orders.id\_order = 2, orders.id\_obat = 2, id\_pasien = 1, tgl\_order = 2023-09-13, harga\_obat = 50000, id\_order\_details = 2, and order\_details.id = 2. The third row shows orders.id\_order = 3, orders.id\_obat = 3, id\_pasien = 2, tgl\_order = 2023-09-28, harga\_obat = 10000, id\_order\_details = <null>, and order\_details.id = <null>.

	orders.id_order	orders.id_obat	id_pasien	tgl_order	harga_obat	id_order_details	order_details.id
1	1	1	2	2023-09-01	20000	1	1
2	2	2	1	2023-09-13	50000	2	2
3	3	3	2	2023-09-28	10000	<null>	<null>

Disini saya melakukan gabungan LEFT dan RIGHT Dengan menggunakan UNION. dimana pada kasus query ini yaitu melakukan LEFT JOIN antara table orders dan orders\_details, kemudian melakukan RIGHT JOIN antara table orders dan order\_details.

hasil query ini akan menampilkan semua kolom pada gabungan dari 2 table tersebut yaitu orders dan orders\_details

## 20. FULL OUTER JOIN Dengan Menggunakan Teknik LEFT JOIN, RIGHT JOIN, UNION, dan COALESCE



The screenshot shows a SQL playground interface. The top panel displays a query in a dark-themed editor. The query is as follows:

```
1 SELECT * FROM orders
2 LEFT JOIN order_details ON orders.id_order = order_details.id_order
3 UNION
4 SELECT * FROM orders
5 RIGHT JOIN order_details ON orders.id_order = order_details.id_order
6 WHERE orders.id_order IS NULL OR order_details.id_order IS NULL;
7
```

The bottom panel shows the output of the query, labeled "Result 12". It displays a table with 7 columns: orders.id\_order, orders.id\_obat, id\_pasien, tgl\_order, harga\_obat, id\_order\_details, and order. The table contains 3 rows of data:

	orders.id_order	orders.id_obat	id_pasien	tgl_order	harga_obat	id_order_details	order
1	1	1	2	2023-09-01	20000	1	
2	2	2	1	2023-09-13	50000	2	
3	3	3	2	2023-09-28	10000	<null>	

yang pertama yaitu melakukan LEFT JOIN antara table orders dan orders\_details tetapi berdasarkan id\_order, pada tahap ini akan menghasilkan dari orders dan order\_details

lalu untuk UNION Sendiri yaitu teknik SQL untuk di gunakan menggabungkan query yang berbeda lebih tepatnya yaitu menggabungkan hasil dari SELECT query yang berbeda

lalu tahap kedua yaitu melakukan RIGHT JOIN antara table orders dan order\_details yang berdasarkan kolom id\_order, di tahap kedua ini akan menghasilkan semua dari order\_details dan yang sesuai dari orders.

tahap terakhir yaitu teknik COALESCE (mengembalikan ekspresi non-null pertama dalam daftar) bagian ini adalah untuk memfilter hasil JOIN untuk ambil dimana orders.id\_order atau order\_details\_order

## 21. SELF JOIN

```
console_1 x
Tx: Auto ✓
Playground
1 ✓ SELECT * FROM orders o1
2 LEFT JOIN order_details od1 ON o1.id_order = od1.id_order
3 UNION
4 SELECT * FROM orders o2
5 RIGHT JOIN order_details od2 ON o2.id_order = od2.id_order
6 WHERE o2.id_order IS NULL OR od2.id_order IS NULL;
7
```

Output Result 14 x

3 rows

	o1.id_order	o1.id_obat	id_pasien	tgl_order	harga_obat	id_order_details	od1.id_order
1	1	1	2	2023-09-01	20000	1	
2	2	2	1	2023-09-13	50000	2	
3	3	3	2	2023-09-28	10000	<null>	<null>