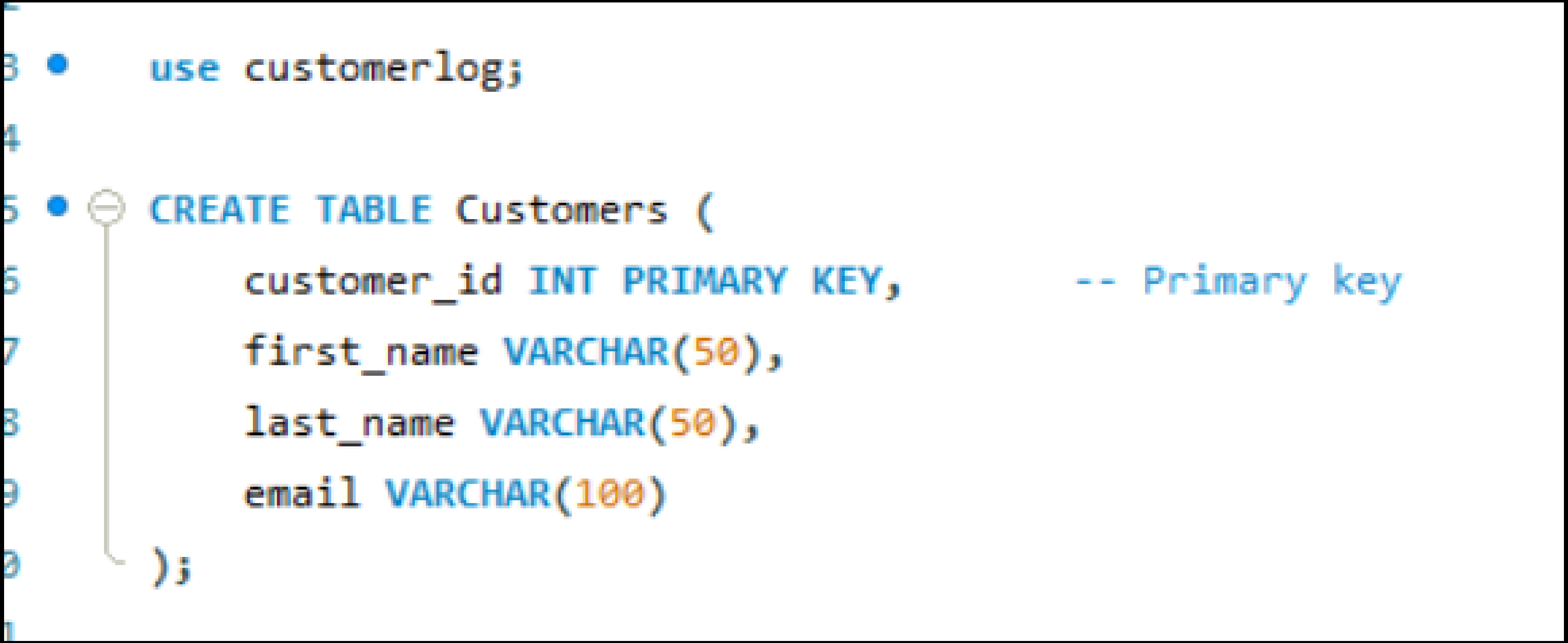
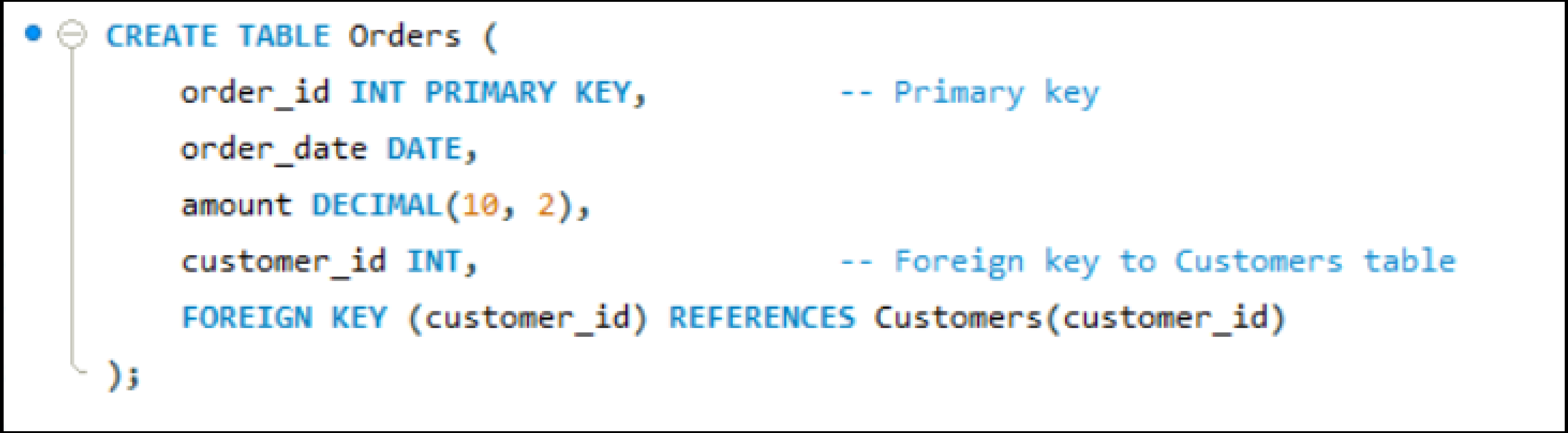
→ Understanding how changing join conditions can help solve complex problems by altering the structure of tables and the resulting outcomes.

Creation of Schema (Customer log) & Table – Customers -DDL commands



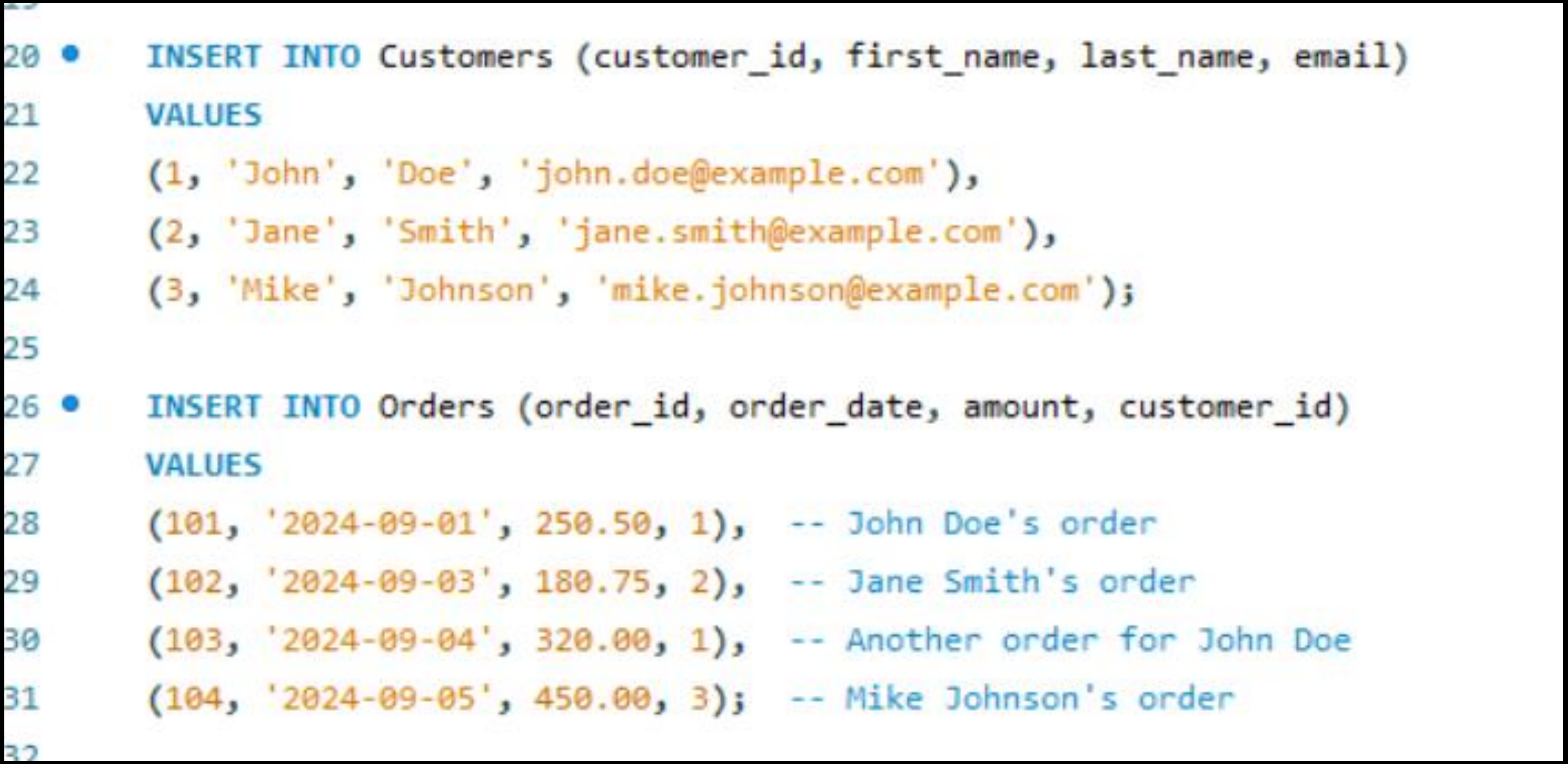
```
use customerlog;

CREATE TABLE Customers (
    customer_id INT PRIMARY KEY,        -- Primary key
    first_name VARCHAR(50),
    last_name VARCHAR(50),
    email VARCHAR(100)
);
```

Creation of orders table



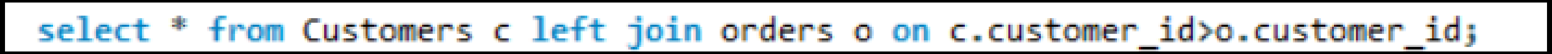
```
CREATE TABLE Orders (
    order_id INT PRIMARY KEY,            -- Primary key
    order_date DATE,
    amount DECIMAL(10, 2),
    customer_id INT,                     -- Foreign key to Customers table
    FOREIGN KEY (customer_id) REFERENCES Customers(customer_id)
);
```

Insertion of values

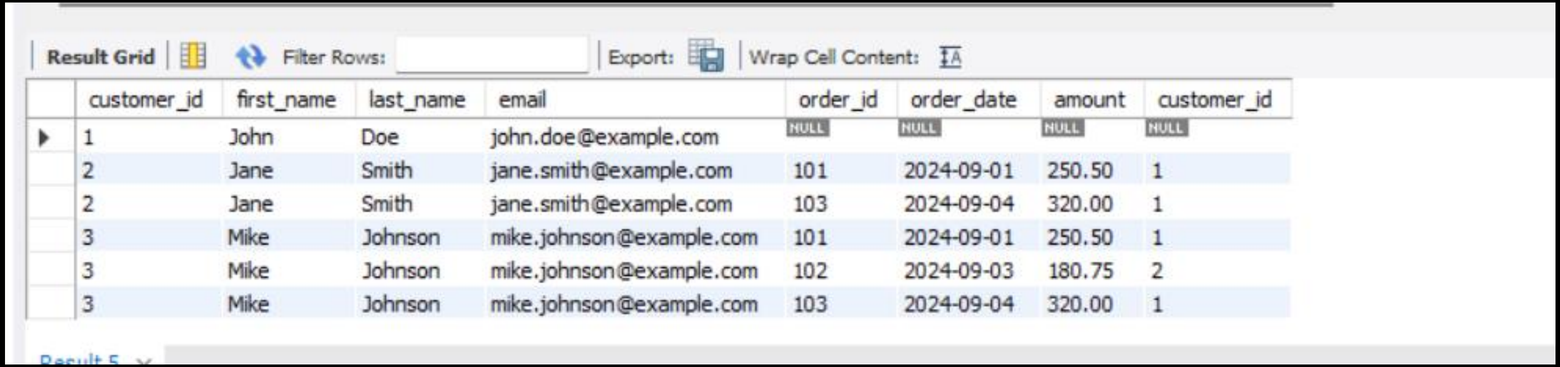


```sql
INSERT INTO Customers (customer_id, first_name, last_name, email)
VALUES
(1, 'John', 'Doe', 'john.doe@example.com'),
(2, 'Jane', 'Smith', 'jane.smith@example.com'),
(3, 'Mike', 'Johnson', 'mike.johnson@example.com');

INSERT INTO Orders (order_id, order_date, amount, customer_id)
VALUES
(101, '2024-09-01', 250.50, 1),  -- John Doe's order
(102, '2024-09-03', 180.75, 2),  -- Jane Smith's order
(103, '2024-09-04', 320.00, 1),  -- Another order for John Doe
(104, '2024-09-05', 450.00, 3);  -- Mike Johnson's order
```

Use joins by changing conditions <> case1:



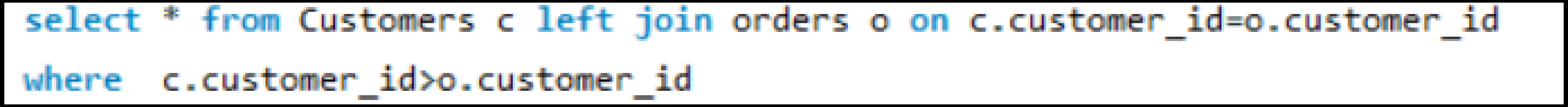
```
select * from Customers c left join orders o on c.customer_id>o.customer_id;
```



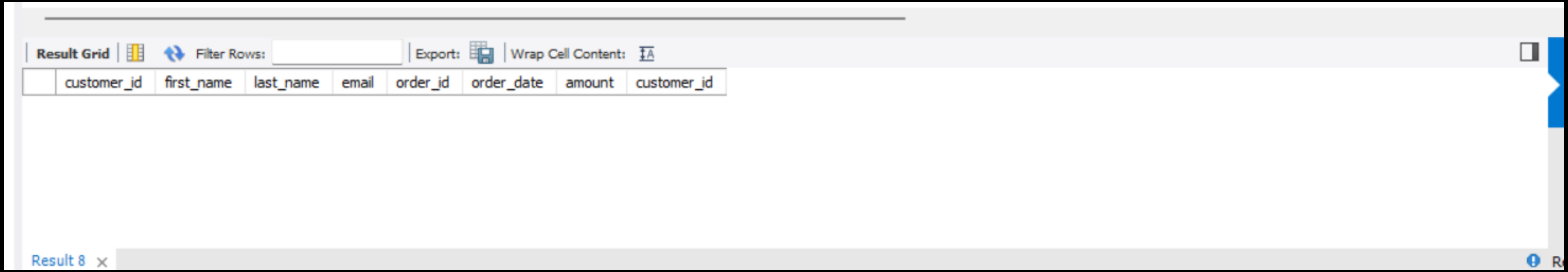
customer_id	first_name	last_name	email	order_id	order_date	amount	customer_id
1	John	Doe	john.doe@example.com	NULL	NULL	NULL	NULL
2	Jane	Smith	jane.smith@example.com	101	2024-09-01	250.50	1
2	Jane	Smith	jane.smith@example.com	103	2024-09-04	320.00	1
3	Mike	Johnson	mike.johnson@example.com	101	2024-09-01	250.50	1
3	Mike	Johnson	mike.johnson@example.com	102	2024-09-03	180.75	2
3	Mike	Johnson	mike.johnson@example.com	103	2024-09-04	320.00	1

In this case, when c.customer_id > o.customer_id is used with a left join, the count of customer_id from the left table remains intact. For rows where the condition is not satisfied, the right table's corresponding entries are filled with null values

Case 2:



```
select * from Customers c left join orders o on c.customer_id=o.customer_id
where  c.customer_id>o.customer_id
```



customer_id	first_name	last_name	email	order_id	order_date	amount	customer_id

In this case, when c.customer_id = o.customer_id is used with a left join, the count of customer_id from the left table remains intact. For rows where the condition is not satisfied, the right table's corresponding entries are filled with null values.Using the logic of order of execution where clause will do filtering and in output We will receive empty table.

- Thanks For Your time !!
- Happy SQling 😊