

SQL QUESTIONS, QUERY AND SCREENSHOTS

Here is the CSV FILE that I have used for solve the questions



customers.csv



orders.csv

TO IMPORT THE DATA IN SQL :

```
create database customers_and_orders;  
  
use customers_and_orders;  
  
CREATE TABLE customers (  
    customer_id VARCHAR(50) PRIMARY KEY,  
    customer_unique_id VARCHAR(50),  
    customer_city VARCHAR(100),  
    customer_state VARCHAR(100)  
);  
  
CREATE TABLE orders (  
    order_id VARCHAR(50) PRIMARY KEY,  
    customer_id VARCHAR(50),  
    order_status VARCHAR(50),  
    order_purchase DATE,  
    order_approved_at DATE,  
    carrier_date DATE,  
    customer_date DATE,  
    delivery_date DATE,  
    FOREIGN KEY (customer_id) REFERENCES customers(customer_id)  
);
```

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server
8.0/Uploads/customers.csv'
INTO TABLE customers
FIELDS TERMINATED BY ','
ENCLOSED BY """
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

```

```

LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/orders.csv'
INTO TABLE orders
FIELDS TERMINATED BY ','
ENCLOSED BY """
LINES TERMINATED BY '\n'
IGNORE 1 ROWS;

```

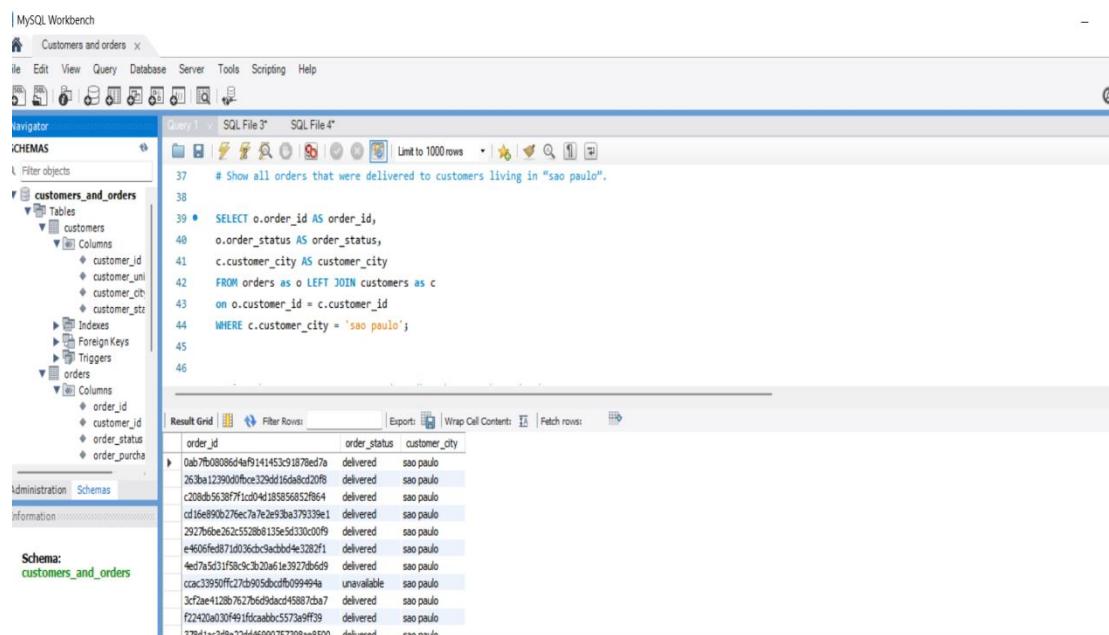
QUERY QUESTIONS AND SCREENSHOTS :

1. Show all orders that were delivered to customers living in "sao paulo".

```

SELECT o.order_id AS order_id,
o.order_status AS order_status,
c.customer_city AS customer_city
FROM orders as o LEFT JOIN customers as c
on o.customer_id = c.customer_id
WHERE c.customer_city = 'sao paulo';

```



The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema `customers_and_orders` with tables `customers` and `orders`.
- Query Editor:** Contains the SQL query:

```

37  # Show all orders that were delivered to customers living in "sao paulo".
38
39  • SELECT o.order_id AS order_id,
40    o.order_status AS order_status,
41    c.customer_city AS customer_city
42  FROM orders as o LEFT JOIN customers as c
43  on o.customer_id = c.customer_id
44  WHERE c.customer_city = 'sao paulo';
45
46

```
- Result Grid:** Displays the results of the query, showing 10 rows of order information where the customer city is 'sao paulo'. The columns are `order_id`, `order_status`, and `customer_city`.

| order_id | order_status | customer_city |
|-------------------------------------|--------------|---------------|
| 0ab7b08086d4bf914153d1878e7a | delivered | sao paulo |
| 263ba12390d0fce329d16da8cd20f8 | delivered | sao paulo |
| c208db5638f7fc0d4d183856852f664 | delivered | sao paulo |
| cd16e89626276ec7a7e2639a3d39339e1 | delivered | sao paulo |
| 29276bb262c5328b135e5d330c0f9 | delivered | sao paulo |
| e460fed87d035cb3acbb04e3282f1 | delivered | sao paulo |
| 4e07d5d1f18c8c2b20a1e3927bd6d9 | delivered | sao paulo |
| czac33950ffcc27d905bdcff09949a | unavailable | sao paulo |
| 3c28e4128b762b6d95acd45887d7a7 | delivered | sao paulo |
| f22420a030f491f5fcabb5573a9ff39 | delivered | sao paulo |
| 778411a749e775144650017c77908aa8f50 | delivered | sao paulo |

2 . List the top 10 most recent orders (based on purchase date).

```
SELECT o.order_id AS order_id,  
o.order_purchase AS purchase_date  
FROM orders as o  
ORDER BY purchase_date  
LIMIT 10;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Navigator' with the 'SCHEMAS' section expanded, showing the 'customers_and_orders' schema with its tables ('customers' and 'orders'). The 'Information' section below it shows the 'customers_and_orders' schema. The main area is titled 'Query 1' and contains the SQL code for listing the top 10 most recent orders. The 'Result Grid' below shows the results, which are empty at this point.

```
File Edit View Query Database Server Tools Scripting Help  
Navigator: Schemas  
SCHEMAS  
Filter objects  
customers_and_orders  
Tables  
customers  
Columns  
customer_id  
customer_uni  
customer_ctn  
customer_stz  
Indexes  
Foreign Keys  
Triggers  
orders  
Columns  
order_id  
customer_id  
order_status  
order_purchase  
Administration Schemas  
Information  
Schema: customers_and_orders  
Result Grid | Filter Rows: | Export/Import: | Wrap Cell Content: | Fetch Rows: |  
order_id purchase_date  
| 2e3d858fb9795e65c102f9d417 2016-09-04  
| 816d7510917f656e0208e4c71925 2016-09-05  
| 809a252a058ca4bd2772efca9a2c 2016-09-13  
| bfb0d98de91302105ad712b648bfec 2016-09-15  
| 71303a7e3b2b399fbad537414edc9fa 2016-10-02  
| be3bc2fb5a149871e3d4f451ad19a9 2016-10-03  
| 65d1e226dafe8bdc42f66542252d14 2016-10-03  
| a41d2759be2ab39ee070303204465 2016-10-03  
| 3b697a2069e427646d2567910af6f657 2016-10-03  
| cf3b8574c82b42f8129f6d502690c3e 2016-10-03  
| null  
orders 27 | Apply
```

3. Find the number of orders placed by each city.

```
SELECT c.customer_city, COUNT(o.order_id) AS total_orders  
FROM orders o  
JOIN customers c ON o.customer_id = c.customer_id  
GROUP BY c.customer_city  
ORDER BY total_orders DESC;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Navigator' with the 'SCHEMAS' section expanded, showing the 'customers_and_orders' schema with its tables ('customers' and 'orders'). The 'Information' section below it shows the 'customers_and_orders' schema. The main area is titled 'Query 1' and contains the SQL code for finding the number of orders placed by each city. The 'Result Grid' below shows the results, which include data for cities like sao paulo, rio de janeiro, belo horizonte, brasilia, curitiba, campinas, porto alegre, sao paulo, quebec, sao bernardo do campo, niteroi, santo andre, osasco, and aracatuba.

```
File Edit View Query Database Server Tools Scripting Help  
Navigator: Schemas  
SCHEMAS  
Filter objects  
customers_and_orders  
Tables  
customers  
Columns  
customer_id  
customer_uni  
customer_ctn  
customer_stz  
Indexes  
Foreign Keys  
Triggers  
orders  
Columns  
order_id  
customer_id  
order_status  
order_purchase  
Administration Schemas  
Information  
Schema: customers_and_orders  
Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Fetch Rows: |  
customer_city total_orders  
| sao paulo 15540  
| rio de janeiro 6882  
| belo horizonte 2773  
| brasilia 2331  
| curitiba 1521  
| campinas 1444  
| porto alegre 1379  
| sao paulo 1189  
| quebec 1245  
| sao bernardo do campo 938  
| niteroi 849  
| santo andre 797  
| osasco 746  
| aracatuba 713  
Result 28 |
```

4 . Show all orders placed after 2024-01-01 and sort by delivery date (latest first).

```
SELECT order_id, order_status, order_purchase, delivery_date
FROM orders
WHERE order_purchase > '2024-01-01'
ORDER BY delivery_date DESC;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'customers_and_orders' selected. The central area contains the SQL editor with the following code:

```
70
71
72
73 # Show all orders placed after 2024-01-01 and sort by delivery date (latest first).
74
75 • SELECT order_id, order_status, order_purchase, delivery_date
76   FROM orders
77   WHERE order_purchase > '2024-01-01'
78   ORDER BY delivery_date DESC;
79
80
81
82
```

The result grid shows one row of data with all columns set to NULL.

5 . Display all orders with customer city and state.

```
SELECT o.order_id, o.order_status, c.customer_city, c.customer_state
FROM orders o
INNER JOIN customers c ON o.customer_id = c.customer_id;
```

The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'Schemas' tree, with 'customers_and_orders' selected. The central area contains the SQL editor with the following code:

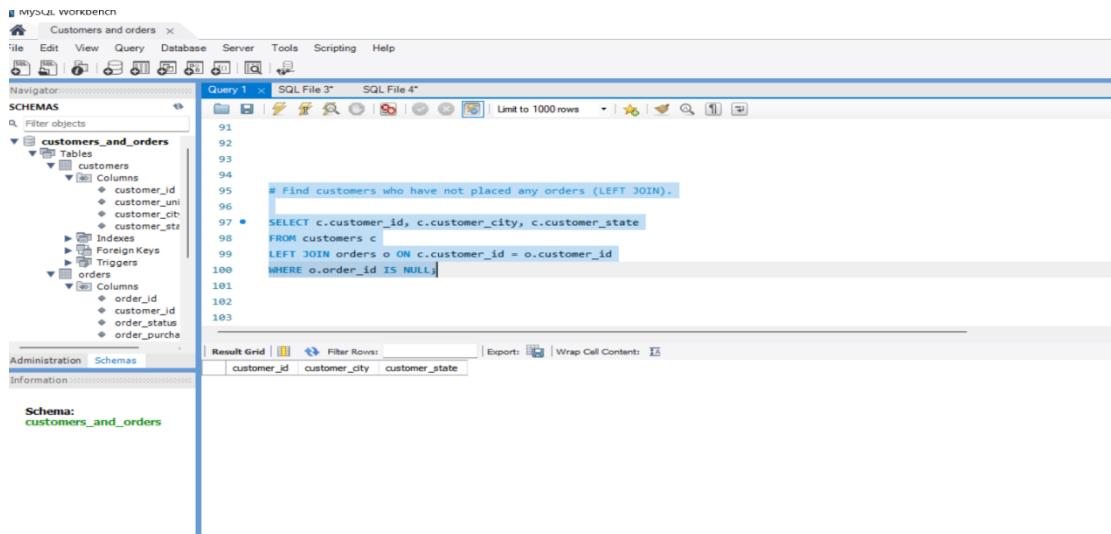
```
82
83
84
85 # Display all orders with customer city and state.
86
87 • SELECT o.order_id, o.order_status, c.customer_city, c.customer_state
88   FROM orders o
89   INNER JOIN customers c ON o.customer_id = c.customer_id;
90
91
92
93
94
```

The result grid displays multiple rows of data, mapping each order to its corresponding customer city and state. The columns are: order_id, order_status, customer_city, and customer_state.

| order_id | order_status | customer_city | customer_state |
|-------------------------------------|--------------|-----------------|----------------|
| 000 10-42febc5ad1ba2dd792cb16214 | delivered | Campinas | RJ |
| 000 18f7772fd320c557190d7a14-8bd3 | delivered | santa fe do sul | SP |
| 000 2295e022-4efcc0d07da4fc70-3e | delivered | para de minas | MG |
| 000 42b26cf59d7ce69dfabbb-4e5b-4f59 | delivered | atibaia | SP |
| 000 42b26cf59d7ce69dfabbb-4e5b-4f59 | delivered | varzea paulista | SP |
| 000 42b26cf59d7ce69dfabbb-4e5b-4f59 | delivered | uberaba | MG |
| 000 42b26cf59d7ce69dfabbb-4e5b-4f59 | delivered | guarulhos | SP |
| 000 576fe3993199472db9d288e5e179e8 | delivered | prado grande | SP |
| 000 5e1a1729c5d78598e2b088904576c | delivered | santos | SP |
| 000 5f5f04142cb953acd1d21e1f9923499 | delivered | jandira | SP |
| 000 5f5f04142cb953acd1d21e1f9923499 | delivered | pindamonhangaba | SP |
| 000 5b381e2406b292ad429470734ebd5 | delivered | monte alto | SP |
| 000 6ec9db01ae64e59a68b2c340bf65a7 | delivered | rio de janeiro | RJ |

6. Find customers who have not placed any orders (LEFT JOIN).

```
SELECT c.customer_id, c.customer_city, c.customer_state  
FROM customers c  
LEFT JOIN orders o ON c.customer_id = o.customer_id  
WHERE o.order_id IS NULL;
```



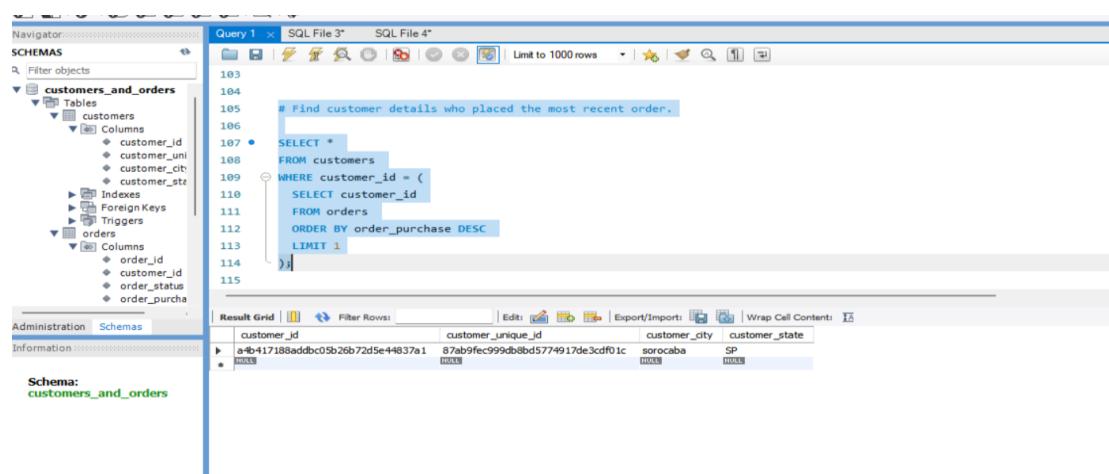
The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'customers_and_orders' schema with its tables ('customers' and 'orders') and columns. The main area contains the SQL query:

```
91  
92  
93  
94  
95 # Find customers who have not placed any orders (LEFT JOIN).  
96  
97 • SELECT c.customer_id, c.customer_city, c.customer_state  
98 FROM customers c  
99 LEFT JOIN orders o ON c.customer_id = o.customer_id  
100 WHERE o.order_id IS NULL;  
101  
102  
103
```

The result grid shows no rows.

7. Find customer details who placed the most recent order.

```
SELECT *  
FROM customers  
WHERE customer_id = (  
    SELECT customer_id  
    FROM orders  
    ORDER BY order_purchase DESC  
    LIMIT 1  
);
```



The screenshot shows the MySQL Workbench interface. The left sidebar displays the 'customers_and_orders' schema with its tables ('customers' and 'orders') and columns. The main area contains the SQL query:

```
103  
104  
105  
106  
107 • # Find customer details who placed the most recent order.  
108 SELECT *  
109 FROM customers  
110 WHERE customer_id = (  
111     SELECT customer_id  
112     FROM orders  
113     ORDER BY order_purchase DESC  
114     LIMIT 1  
115 );
```

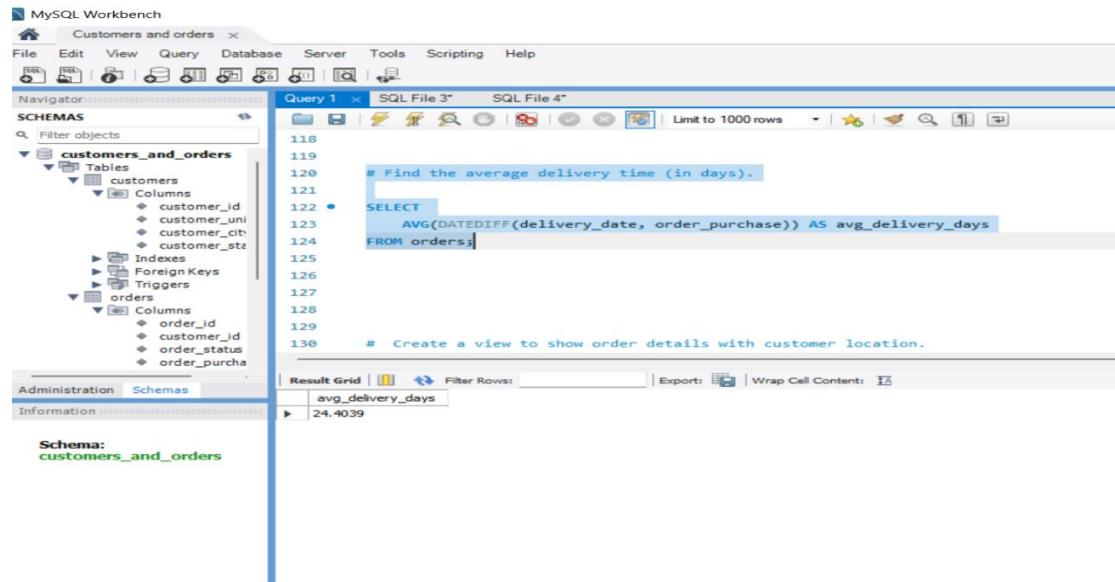
The result grid shows one row of data:

| customer_id | customer_unique_id | customer_city | customer_state |
|----------------------------------|----------------------------------|---------------|----------------|
| a4b417188addbc05b26b72d5e44837a1 | 87ab9fec999db8bd5774917de3cdf01c | sorocaba | SP |
| NULL | NULL | NULL | NULL |

8. Find the average delivery time (in days).

SELECT

```
AVG(DATEDIFF(delivery_date, order_purchase)) AS avg_delivery_days  
FROM orders;
```



The screenshot shows the MySQL Workbench interface. On the left, the Navigator pane displays the schema 'customers_and_orders' with its tables ('customers', 'orders') and columns. The main area, 'Query 1', contains the SQL query for finding the average delivery time. The result grid at the bottom shows a single row with the value '24.4039'.

| avg_delivery_days |
|-------------------|
| 24.4039 |

9. Create a view to show order details with customer location.

VIEWS

```
CREATE VIEW order_customer_summary AS  
SELECT  
    o.order_id,  
    o.order_status,  
    o.delivery_date,  
    c.customer_city,  
    c.customer_state  
FROM orders o  
JOIN customers c ON o.customer_id = c.customer_id;
```

QUERY

```
SHOW FULL TABLES WHERE TABLE_TYPE = 'VIEW';
```

```
SELECT * FROM order_customer_summary LIMIT 10;
```

```
-- Filter by city
```

```
SELECT * FROM order_customer_summary
```

WHERE customer_city = 'sao paulo';

-- Count orders per state

```
SELECT customer_state, COUNT(*) AS total_orders
FROM order_customer_summary
GROUP BY customer_state;
```

The screenshot shows the MySQL Workbench interface. In the left sidebar, under the 'SCHEMAS' section, the 'customers_and_orders' schema is selected. The 'Tables' section contains two tables: 'customers' and 'orders'. The 'customers' table has columns: customer_id, customer_uni, customer_cit, and customer_stt. The 'orders' table has columns: order_id, customer_id, order_status, and order_purcha. Below the tables are sections for 'Indexes', 'Foreign Keys', and 'Triggers'. The main area shows a query editor with the following code:

```
120 # Find the average delivery time (in days).
121
122 • SELECT
123     AVG(DATEDIFF(delivery_date, order_purchase)) AS avg_delivery_days
124     FROM orders;
125
126
127
128 # Create a view to show order details with customer location.
129
130 • SHOW FULL TABLES WHERE TABLE_TYPE = 'VIEW';
131
132 • SELECT * FROM order_customer_summary LIMIT 10;
133
```

Below the code is a 'Result Grid' table with columns: order_id, order_status, delivery_date, customer_cit, and customer_state. The data includes various cities like campos dos goytacazes, santo aleixo, para de minas, etc., and states like RJ, SP, MG, etc.

The screenshot shows the MySQL Workbench interface with the same schema and table structure. The main area shows a query editor with the following code:

```
131
132 -- Filter by city
133 • SELECT * FROM order_customer_summary
134     WHERE customer_city = 'sao paulo';
135
136 -- Count orders per state
137
138 • SELECT customer_state, COUNT(*) AS total_orders
139     FROM order_customer_summary
140     GROUP BY customer_state;
141
142
143
144
145
```

Below the code is a 'Result Grid' table with columns: customer_state and total_orders. The data shows the count of orders for each state where the customer city is 'sao paulo', such as AC (81), AL (413), AM (148), AP (68), BA (3300), CE (1336), DF (2140), ES (2033), GO (2020), MA (747), MG (14235), MS (715), MT (907), PA (575), PR (576).

The screenshot shows the MySQL Workbench interface with the same schema and table structure. The main area shows a query editor with the following code:

```
130
131
132 -- Show full tables
133 • SHOW FULL TABLES WHERE TABLE_TYPE = 'VIEW';
134
135 • SELECT * FROM order_customer_summary
136     WHERE customer_city = 'sao paulo';
137
138 -- Count orders per state
139
140 • SELECT customer_state, COUNT(*) AS total_orders
141     FROM order_customer_summary
142     GROUP BY customer_state;
143
```

Below the code is a 'Result Grid' table with columns: order_id, order_status, delivery_date, customer_cit, and customer_state. The data includes various cities like sao paulo, santo aleixo, para de minas, etc., and states like SP, MG, etc.

10 Add indexes for faster joins and filtering.

INDEX

```
CREATE INDEX idx_customer_id ON orders(customer_id);
```

```
CREATE INDEX idx_customer_state ON customers(customer_state);
```

QUERY

The screenshot shows the MySQL Workbench interface with two queries in the Query Editor:

```
-- Count orders per state
SELECT customer_state, COUNT(*) AS total_orders
FROM order_customer_summary
GROUP BY customer_state;
```

```
-- SHOW INDEX FROM customers;
SHOW INDEX FROM orders;
```

The Result Grid shows the following index details:

| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
|-----------|------------|--------------------|--------------|----------------|-----------|-------------|----------|--------|------|------------|---------|---------------|---------|------------|
| customers | 0 | PRIMARY | 1 | customer_id | A | 103008 | | | | BTREE | | YES | HOLLOW | |
| customers | 1 | idx_customer_state | 1 | customer_state | A | 27 | | | YES | BTREE | | YES | HOLLOW | |

The screenshot shows the MySQL Workbench interface with two queries in the Query Editor:

```
-- Count orders per state
SELECT customer_state, COUNT(*) AS total_orders
FROM order_customer_summary
GROUP BY customer_state;
```

```
-- SHOW INDEX FROM customers;
SHOW INDEX FROM orders;
```

The Result Grid shows the following index details:

| Table | Non_unique | Key_name | Seq_in_index | Column_name | Collation | Cardinality | Sub_part | Packed | Null | Index_type | Comment | Index_comment | Visible | Expression |
|--------|------------|-----------------|--------------|-------------|-----------|-------------|----------|--------|------|------------|---------|---------------|---------|------------|
| orders | 0 | PRIMARY | 1 | order_id | A | 106875 | | | | BTREE | | YES | HOLLOW | |
| orders | 1 | idx_customer_id | 1 | customer_id | A | 98679 | | | YES | BTREE | | YES | HOLLOW | |

