

Data Science Bootcamp

Homework 4

Customer Table:

Programiz
Online SQL Editor

Interactive SQL Course

CUSTOMERS [-]

Customer_id [integer]

First_name [text]

Last_name [text]

Address [text]

Input

Run SQL

Available Tables

CUSTOMERS

Customer_id	First_name	Last_name	Address
1	John	Doe	NY
2	Alice	Smith	CA
3	Bob	Johnson	TX
4	Jane	Brown	WA
5	Charlie	Davis	NV
6	John	Doe	NJ

SQL query successfully executed. However, the result set is empty.

Items Table:

Programiz

Online SQL Editor

Interactive SQL Course

CUSTOMERS [-]

Customer_id [integer]

First_name [text]

Last_name [text]

Address [text]

ITEMS [-]

Item_id [integer]

Item_name [text]

Price [real]

Department [text]

<

Input

>

Run SQL

>

Available Tables

```
CREATE TABLE ITEMS (
  Item_id INTEGER PRIMARY KEY,
  Item_name TEXT,
  Price REAL,
  Department TEXT
);

INSERT INTO ITEMS (Item_id, Item_name, Price, Department) VALUES
(1, 'Mouse', 100, 'Electronics'),
(2, 'Keyboard', 150, 'Electronics'),
(3, 'Water Bottle', 100, 'Lifestyle'),
(4, 'Notebook', 200, 'Stationery'),
(5, 'T-shirt', 150, 'Apparel');
```

Output

SQL query successfully executed. However, the result set is empty.

CUSTOMERS

Customer_id	First_name	Last_name	Address
1	John	Doe	NY
2	Alice	Smith	CA
3	Bob	Johnson	TX
4	Jane	Brown	WA
5	Charlie	Davis	NV
6	John	Doe	NJ

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

Sales Table:

Programiz

Online SQL Editor

Interactive SQL Course

CUSTOMERS [-]

Customer_id [integer]

First_name [text]

Last_name [text]

Address [text]

ITEMS [-]

Item_id [integer]

Item_name [text]

Price [real]

Department [text]

SALES [-]

Date [date]

Order_id [integer]

Item_id [integer]

Customer_id [integer]

Quantity [integer]

Revenue [real]

<

Input

>

Run SQL

>

Available Tables

```
CREATE TABLE SALES (
  Date DATE,
  Order_id INTEGER,
  Item_id INTEGER,
  Customer_id INTEGER,
  Quantity INTEGER,
  Revenue REAL,
  FOREIGN KEY (Item_id) REFERENCES ITEMS(Item_id),
  FOREIGN KEY (Customer_id) REFERENCES CUSTOMERS(Customer_id)
);

INSERT INTO SALES (Date, Order_id, Item_id, Customer_id, Quantity, Revenue) VALUES
('2023-03-18', 101, 1, 1, 2, 200),
('2023-03-18', 102, 2, 2, 1, 150),
('2023-01-05', 103, 3, 3, 3, 300),
('2023-01-15', 104, 1, 2, 1, 100),
('2022-12-20', 105, 4, 4, 2, 400),
('2022-06-15', 106, 5, 5, 1, 150),
('2023-03-18', 107, 1, 6, 2, 200);
```

Output

SQL query successfully executed. However, the result set is empty.

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity	Rev
2023-03-18	101	1	1	2	200
2023-03-18	102	2	2	1	150
2023-01-05	103	3	3	3	300
2023-01-15	104	1	2	1	100
2022-12-20	105	4	4	2	400
2022-06-15	106	5	5	1	150
2023-03-18	107	1	6	2	200

Pull total number of orders that were completed on 18th March 2023

- Pull total number of orders that were completed on 18th March 2023 with the first name 'John' and last name Doe'

[View SQL Query](#)

Input

```
SELECT COUNT(DISTINCT s.Order_id) AS john_doe_orders
FROM SALES s
JOIN CUSTOMERS c ON s.Customer_id = c.Customer_id
WHERE s.Date = '2023-03-18'
      AND c.first_name = 'John'
      AND c.last_name = 'Doe';
```

Run SQL

Available Tables

Output

john_doe_orders
2

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity	Rev
2023-03-18	101	1	1	2	200
2023-03-18	102	2	2	1	150
2023-01-05	103	3	3	3	300
2023-01-15	104	1	2	1	100
2022-12-20	105	4	4	2	400
2022-06-15	106	5	5	1	150
2023-03-18	107	1	6	2	200

- Pull total number of customers that purchased in January 2023 and the average amount spend per customer

Programiz
Online SQL Editor

Interactive SQL

CUSTOMERS [-]

- Customer_id [integer]
- First_name [text]
- Last_name [text]
- Address [text]

ITEMS [-]

- Item_id [integer]
- Item_name [text]
- Price [real]
- Department [text]

SALES [-]

- Date [date]
- Order_id [integer]
- Item_id [integer]
- Customer_id [integer]
- Quantity [integer]
- Revenue [real]

Input

```

SELECT
  COUNT(DISTINCT Customer_id) AS total_customers,
  AVG(customer_total) AS avg_spent_per_customer
FROM (
  SELECT Customer_id, SUM(Revenue) AS customer_total
  FROM SALES
  WHERE Date >= '2023-01-01' AND Date < '2023-02-01'
  GROUP BY Customer_id
) sub;

```

Run SQL

Available Tables

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity
2023-03-18	101	1	1	2
2023-03-18	102	2	2	1
2023-01-05	103	3	3	3
2023-01-15	104	1	2	1
2022-12-20	105	4	4	2
2022-06-15	106	5	5	1
2023-03-18	107	1	6	2

Output

total_customers	avg_spent_per_customer
2	200

- Pull the departments that generated less than \$600 in 2022

Programiz
Online SQL Editor

Interactive SQL Course

CUSTOMERS [-]

- Customer_id [integer]
- First_name [text]
- Last_name [text]
- Address [text]

ITEMS [-]

- Item_id [integer]
- Item_name [text]
- Price [real]
- Department [text]

SALES [-]

- Date [date]
- Order_id [integer]
- Item_id [integer]
- Customer_id [integer]
- Quantity [integer]
- Revenue [real]

Input

```

SELECT i.Department, SUM(s.Revenue) AS total_revenue
FROM SALES s
JOIN ITEMS i ON s.Item_id = i.Item_id
WHERE s.Date >= '2022-01-01' AND s.Date < '2023-01-01'
GROUP BY i.Department
HAVING SUM(s.Revenue) < 600;

```

Run SQL

Available Tables

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity	Rev
2023-03-18	101	1	1	2	200
2023-03-18	102	2	2	1	150
2023-01-05	103	3	3	3	300
2023-01-15	104	1	2	1	100
2022-12-20	105	4	4	2	400
2022-06-15	106	5	5	1	150
2023-03-18	107	1	6	2	200

Output

Department	total_revenue
Apparel	150
Stationery	400

- What is the most and least revenue we have generated by an order

Programiz

Online SQL Editor

Interactive SQL Course

CUSTOMERS [-]

Customer_id [integer]

First_name [text]

Last_name [text]

Address [text]

ITEMS [-]

Item_id [integer]

Item_name [text]

Price [real]

Department [text]

SALES [-]

Date [date]

Order_id [integer]

Item_id [integer]

Customer_id [integer]

Quantity [integer]

Revenue [real]

Input

```
SELECT
  MAX(order_total) AS max_revenue,
  MIN(order_total) AS min_revenue
FROM (
  SELECT Order_id, SUM(Revenue) AS order_total
  FROM SALES
  GROUP BY Order_id
) sub;
```

Run SQL

Available Tables

Output

max_revenue	min_revenue
400	100

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity	Rev
2023-03-18	101	1	1	2	200
2023-03-18	102	2	2	1	150
2023-01-05	103	3	3	3	300
2023-01-15	104	1	2	1	100
2022-12-20	105	4	4	2	400
2022-06-15	106	5	5	1	150
2023-03-18	107	1	6	2	200

- What were the orders that were purchased in our most lucrative order

CUSTOMERS [-]

Customer_id [integer]

First_name [text]

Last_name [text]

Address [text]

ITEMS [-]

Item_id [integer]

Item_name [text]

Price [real]

Department [text]

SALES [-]

Date [date]

Order_id [integer]

Item_id [integer]

Customer_id [integer]

Quantity [integer]

Revenue [real]

Input

```
WITH order_revenue AS (  
  SELECT Order_id, SUM(Revenue) AS total_revenue  
  FROM SALES  
  GROUP BY Order_id  
)  
SELECT s.*  
FROM SALES s  
JOIN order_revenue o ON s.Order_id = o.Order_id  
WHERE o.total_revenue = (  
  SELECT MAX(total_revenue) FROM order_revenue  
);
```

Output

Date	Order_id	Item_id	Customer_id	Quantity	Revenue
2022-12-20	105	4	4	2	400

Available Tables

ITEMS

Item_id	Item_name	Price	Department
1	Mouse	100	Electronics
2	Keyboard	150	Electronics
3	Water Bottle	100	Lifestyle
4	Notebook	200	Stationery
5	T-shirt	150	Apparel

SALES

Date	Order_id	Item_id	Customer_id	Quantity	Rev
2023-03-18	101	1	1	2	200
2023-03-18	102	2	2	1	150
2023-01-05	103	3	3	3	300
2023-01-15	104	1	2	1	100
2022-12-20	105	4	4	2	400
2022-06-15	106	5	5	1	150
2023-03-18	107	1	6	2	200