

Ruchi Jha

The table-

```
CREATE TABLE CUSTOMERS (  
    customer_id INT PRIMARY KEY,  
    first_name VARCHAR(50),  
    last_name VARCHAR(50),  
    address VARCHAR(100)  
);
```

```
CREATE TABLE ITEMS (  
    item_id INT PRIMARY KEY,  
    item_name VARCHAR(100),  
    price DECIMAL(10, 2),  
    department VARCHAR(50)  
);
```

```
CREATE TABLE SALES (  
    date DATE,  
    order_id INT PRIMARY KEY,  
    item_id INT,  
    customer_id INT,  
    quantity INT,  
    revenue DECIMAL(10, 2),  
    FOREIGN KEY (item_id) REFERENCES ITEMS(item_id),  
    FOREIGN KEY (customer_id) REFERENCES CUSTOMERS(customer_id)  
);
```

Filling in values-

-- CUSTOMERS

INSERT INTO CUSTOMERS VALUES

```
(1, 'John', 'Doe', '101 Elm St'),  
(2, 'Alice', 'Smith', '202 Oak Ave'),  
(3, 'Bob', 'Brown', '303 Pine Rd'),  
(4, 'Mary', 'Johnson', '404 Maple Blvd');
```

-- ITEMS

INSERT INTO ITEMS VALUES

```
(201, 'Laptop', 1200.00, 'Electronics'),  
(202, 'Notebook', 5.00, 'Stationery'),  
(203, 'Coffee Mug', 15.00, 'Kitchenware'),  
(204, 'Wireless Mouse', 25.00, 'Electronics'),  
(205, 'Desk Lamp', 45.00, 'Home Decor');
```

-- SALES

INSERT INTO SALES VALUES

```
('2023-03-18', 3001, 201, 1, 1, 1200.00), -- John Doe
('2023-03-18', 3002, 202, 2, 10, 50.00), -- Alice
('2023-03-18', 3003, 203, 3, 2, 30.00), -- Bob
('2023-01-15', 3004, 204, 1, 1, 25.00), -- John Jan purchase
('2023-01-21', 3005, 201, 4, 1, 1200.00), -- Mary Jan purchase
('2022-07-10', 3006, 203, 2, 2, 30.00), -- Kitchenware < $600
('2022-10-05', 3007, 202, 3, 5, 25.00), -- Stationery < $600
('2023-03-25', 3008, 205, 1, 2, 90.00), -- Home Decor
('2023-03-30', 3009, 204, 2, 1, 25.00), -- Low revenue order
('2023-03-31', 3010, 201, 4, 2, 2400.00); -- Highest revenue order
```

The tables-

The screenshot shows a database management application interface. On the left, a sidebar lists database connections: "Add DataBase", SQLite (selected), MariaDB, PostgreSQL, and MS SQL. The main area displays a SQLite database with three tables: CUSTOMERS, ITEMS, and SALES. The CUSTOMERS table is expanded, showing a list of records with columns: customer_id, first_name, last_name, and address. The records are: (1, 'John', 'Doe', '101 Elm St'), (2, 'Alice', 'Smith', '202 Oak Ave'), (3, 'Bob', 'Brown', '303 Pine Rd'), and (4, 'Mary', 'Johnson', '404 Maple Blvd'). The right-hand panel shows a history of SQL queries executed, including a SELECT query for CUSTOMERS, an INSERT query for CUSTOMERS, and a CREATE TABLE query for CUSTOMERS.

customer_id	first_name	last_name	address
1	John	Doe	101 Elm St
2	Alice	Smith	202 Oak Ave
3	Bob	Brown	303 Pine Rd
4	Mary	Johnson	404 Maple Blvd

Pricing Help + Import Export

+ → "Add DataBase"

Create a database linked to your account.

This is only available with a paid subscription.

SQLite 0.1.4 beta (Mem...)

Table

- CUSTOMERS
- ITEMS
- SALES

MariaDB

PostgreSQL

MS SQL

Run SQLite SQLite.12 SQLite.13

```

1 -- Print all records from the CUSTOMERS table
2 SELECT * FROM CUSTOMERS;
3
4 -- Print all records from the ITEMS table
5 SELECT * FROM ITEMS;
6
7 -- Print all records from the SALES table
8 SELECT * FROM SALES;
9

```

item_id	item_name	price	department
201	Laptop	1200	Electronics
202	Notebook	5	Stationery
203	Coffee Mug	15	Kitchenware
204	Wireless Mouse	25	Electronics
205	Desk Lamp	45	Home Decor

History

Syntax History

SQLite.13

```

-- Print all records from the CUSTOMERS table
SELECT * FROM CUSTOMERS;

-- Print all records from the SALES table
SELECT * FROM SALES;

```

01:18:41

SQLite.12

```

-- CUSTOMERS
INSERT INTO CUSTOMERS VALUES
(1, 'John', 'Doe', '101 Elm St'),
(2, 'Alice', 'Smith', '2 ...

```

01:18:23

SQLite

```

CREATE TABLE CUSTOMERS (
  customer_id INT PRIMARY KEY,
  first_name VARCHAR(50),
  last_name ...

```

01:18:17

Pricing Help + Import Export

+ → "Add DataBase"

Create a database linked to your account.

This is only available with a paid subscription.

SQLite 0.1.4 beta (Mem...)

Table

- CUSTOMERS
- ITEMS
- SALES

MariaDB

PostgreSQL

MS SQL

Run SQLite SQLite.12 SQLite.13

```

1 -- Print all records from the CUSTOMERS table
2 SELECT * FROM CUSTOMERS;
3
4 -- Print all records from the ITEMS table
5 SELECT * FROM ITEMS;
6
7 -- Print all records from the SALES table
8 SELECT * FROM SALES;
9

```

date	order_id	item_id	customer_id	quantity	revenue
2023-03-18	3001	201	1	1	1200
2023-03-18	3002	202	2	10	50
2023-03-18	3003	203	3	2	30
2023-01-15	3004	204	1	1	25
2023-01-21	3005	201	4	1	1200
2022-07-10	3006	203	2	2	30
2022-10-05	3007	202	3	5	25
2023-03-25	3008	205	1	2	90
2023-03-30	3009	204	2	1	25
2023-03-31	3010	201	4	2	2400

History

Syntax History

SQLite.13

```

-- Print all records from the CUSTOMERS table
SELECT * FROM CUSTOMERS;

-- Print all records from the SALES table
SELECT * FROM SALES;

```

01:18:41

SQLite.12

```

-- CUSTOMERS
INSERT INTO CUSTOMERS VALUES
(1, 'John', 'Doe', '101 Elm St'),
(2, 'Alice', 'Smith', '2 ...

```

01:18:23

SQLite

```

CREATE TABLE CUSTOMERS (
  customer_id INT PRIMARY KEY,
  first_name VARCHAR(50),
  last_name ...

```

01:18:17

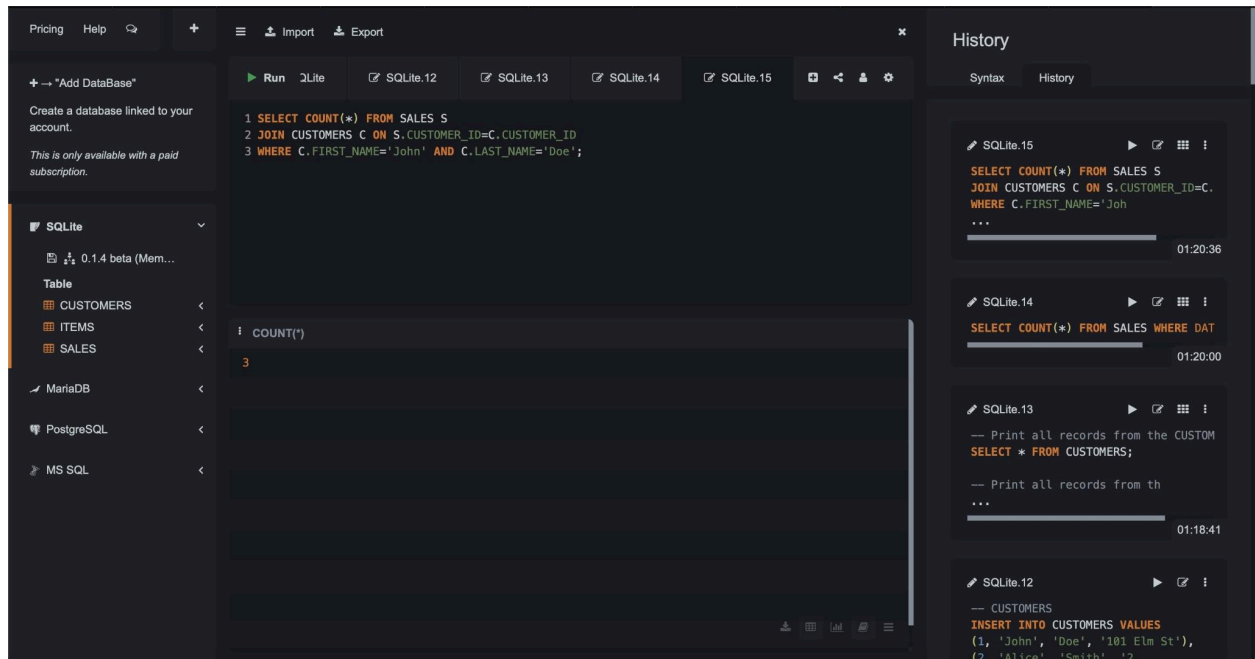
Question 1.Pull total number of orders that were completed on 18th March 2023 with the first name 'John' and last name Doe'

Code-

```

SELECT COUNT(*) FROM SALES S
JOIN CUSTOMERS C ON S.CUSTOMER_ID=C.CUSTOMER_ID
WHERE C.FIRST_NAME='John' AND C.LAST_NAME='Doe';

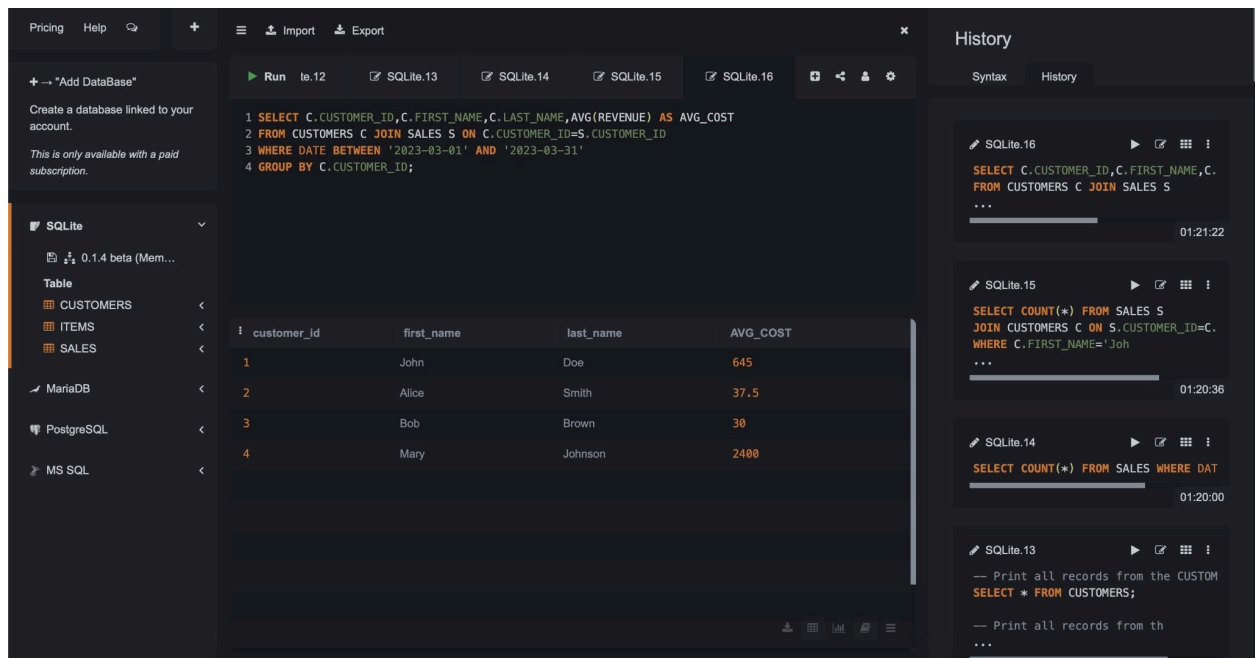
```



Question 2.Pull total number of customers that purchased in January 2023 and the average amount spend per customer

Code-

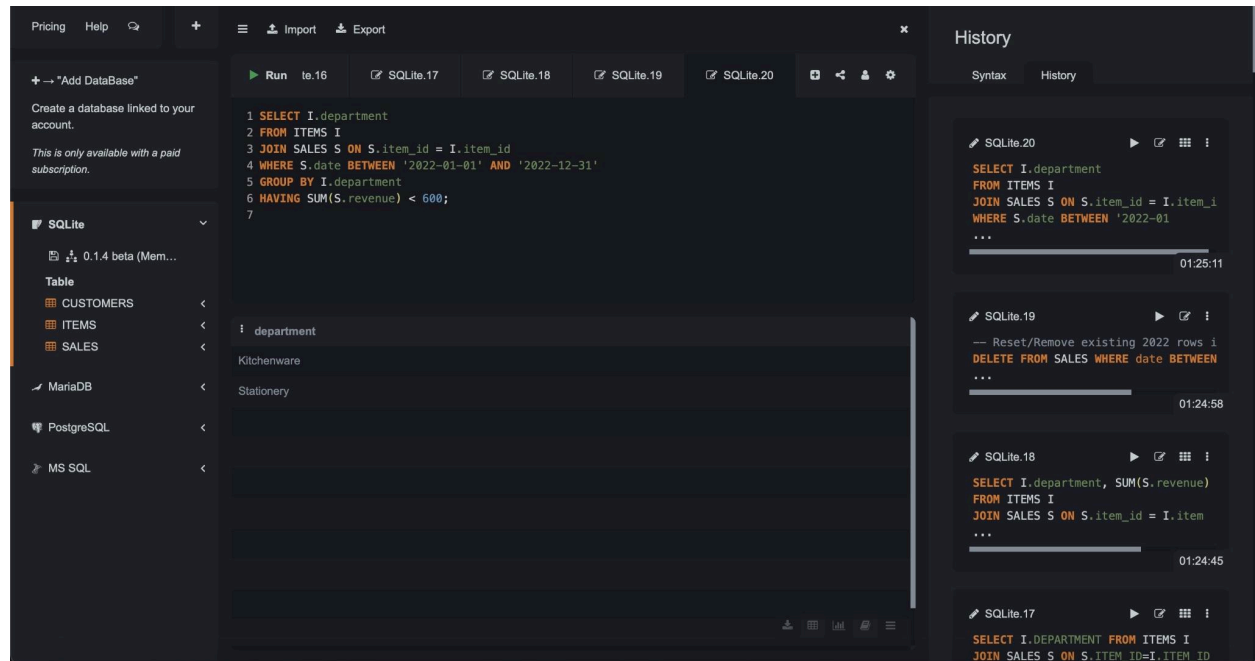
```
SELECT C.CUSTOMER_ID,C.FIRST_NAME,C.LAST_NAME,AVG(REVENUE) AS AVG_COST
FROM CUSTOMERS C JOIN SALES S ON C.CUSTOMER_ID=S.CUSTOMER_ID
WHERE DATE BETWEEN '2023-03-01' AND '2023-03-31'
GROUP BY C.CUSTOMER_ID;
```



Question 3-Pull the departments that generated less than \$600 in 2022

Code-

```
SELECT I.DEPARTMENT
FROM ITEMS I
JOIN SALES S ON S.ITEM_ID=I.ITEM_ID
WHERE DATE BETWEEN '2022-01-01' AND '2022-12-31'
GROUP BY I.DEPARTMENT
HAVING SUM(REVENUE)<600;
```



Question 4-What is the most and least revenue we have generated by an order

Code-

```
SELECT MAX(REVENUE) AS MAX_REVENUE,MIN(REVENUE) AS MIN_REVENUE
FROM SALES;
```

Pricing Help + Import Export

+ → "Add DataBase"

Create a database linked to your account.
This is only available with a paid subscription.

SQLite 0.1.4 beta (Mem...)

Table

- CUSTOMERS
- ITEMS
- SALES

MariaDB

PostgreSQL

MS SQL

Run te.17 SQLite.18 SQLite.19 SQLite.20 SQLite.21

```
1 SELECT MAX(REVENUE) AS MAX_REVENUE, MIN(REVENUE) AS MIN_REVENUE
2 FROM SALES;
```

MAX_REVENUE	MIN_REVENUE
2400	25

History

Syntax History

SQLite.21

```
SELECT MAX(REVENUE) AS MAX_REVENUE, MIN_REVENUE
FROM SALES;
```

01:25:45

SQLite.20

```
SELECT I.department
FROM ITEMS I
JOIN SALES S ON S.item_id = I.item_id
WHERE S.date BETWEEN '2022-01'
...
```

01:25:11

SQLite.19

```
-- Reset/Remove existing 2022 rows i
DELETE FROM SALES WHERE date BETWEEN
...
```

01:24:58

SQLite.18

```
SELECT I.department, SUM(S.revenue)
FROM ITEMS I
JOIN SALES S ON S.item_id = I.item_id
...
```

Question 6-What were the orders that were purchased in our most lucrative order

Code-

SELECT * FROM SALES

WHERE REVENUE=(SELECT MAX(REVENUE) FROM SALES);

Pricing Help + Import Export

+ → "Add DataBase"

Create a database linked to your account.
This is only available with a paid subscription.

SQLite 0.1.4 beta (Mem...)

Table

- CUSTOMERS
- ITEMS
- SALES

MariaDB

PostgreSQL

MS SQL

Run te.18 SQLite.19 SQLite.20 SQLite.21 SQLite.22

```
1 SELECT * FROM SALES
2 WHERE REVENUE=(SELECT MAX(REVENUE) FROM SALES);
```

date	order_id	item_id	customer_id	quantity	revenue
2023-03-31	3010	201	4	2	2400

History

Syntax History

SQLite.22

```
SELECT * FROM SALES
WHERE REVENUE=(SELECT MAX(REVENUE) FROM SALES);
```

01:26:24

SQLite.21

```
SELECT MAX(REVENUE) AS MAX_REVENUE, MIN_REVENUE
FROM SALES;
```

01:25:45

SQLite.20

```
SELECT I.department
FROM ITEMS I
JOIN SALES S ON S.item_id = I.item_id
WHERE S.date BETWEEN '2022-01'
...
```

01:25:11

SQLite.19

```
-- Reset/Remove existing 2022 rows i
DELETE FROM SALES WHERE date BETWEEN
...
```

01:24:58