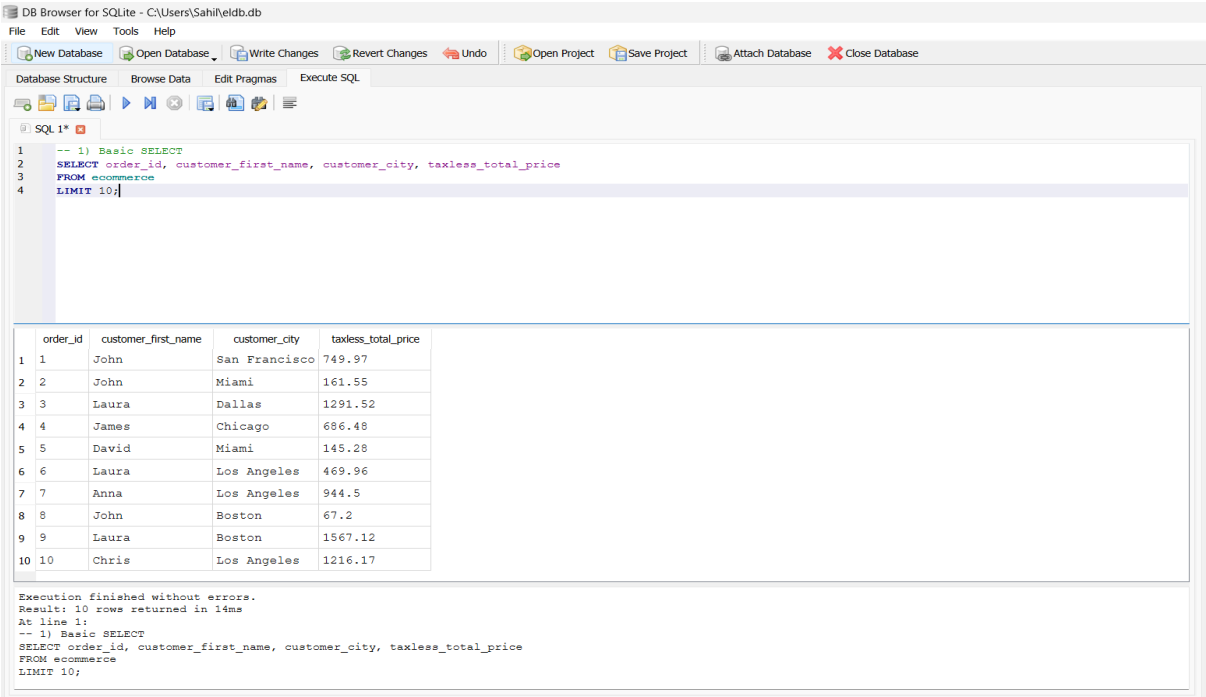
Elevate Labs Internship Task 4 [by – Sahil Wasta]

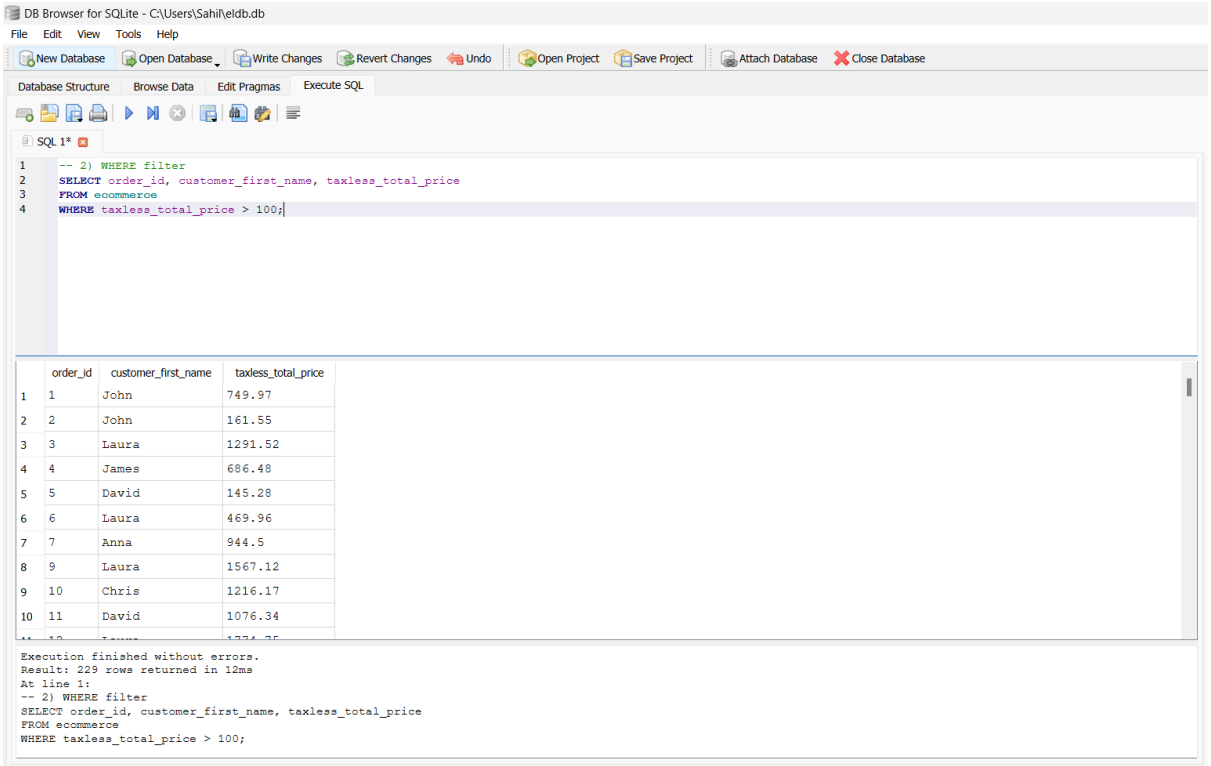
SQL for Data Analysis (Query Screenshots & Insights)

Dataset : Ecommerce Dataset

1) Basic SELECT query to retrieve the data of first 10 rows :



2) WHERE Filter query to retrieve the data where taxless_total_price > 100 :



3) ORDER BY query to retrieve data as ordered in descending order on field taxless_total_price :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 3) ORDER BY
SELECT order_id, customer_first_name, taxless_total_price
FROM ecommerce
ORDER BY taxless_total_price DESC;
```

	order_id	customer_first_name	taxless_total_price
1	133	Michael	2472.75
2	220	Jane	2454.4
3	194	Anna	2412.65
4	115	Chris	2402.45
5	70	Laura	2324.65
6	191	James	2293.55
7	169	David	2268.6
8	15	Emily	2255.55
9	34	Anna	2225.2
10	209	Emily	2141.65

```
Execution finished without errors.
Result: 250 rows returned in 13ms
At line 1:
-- 3) ORDER BY
SELECT order_id, customer_first_name, taxless_total_price
FROM ecommerce
ORDER BY taxless_total_price DESC;
```

4)GROUP BY query to calculate total sales for each city & sort them in descending order of sales :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

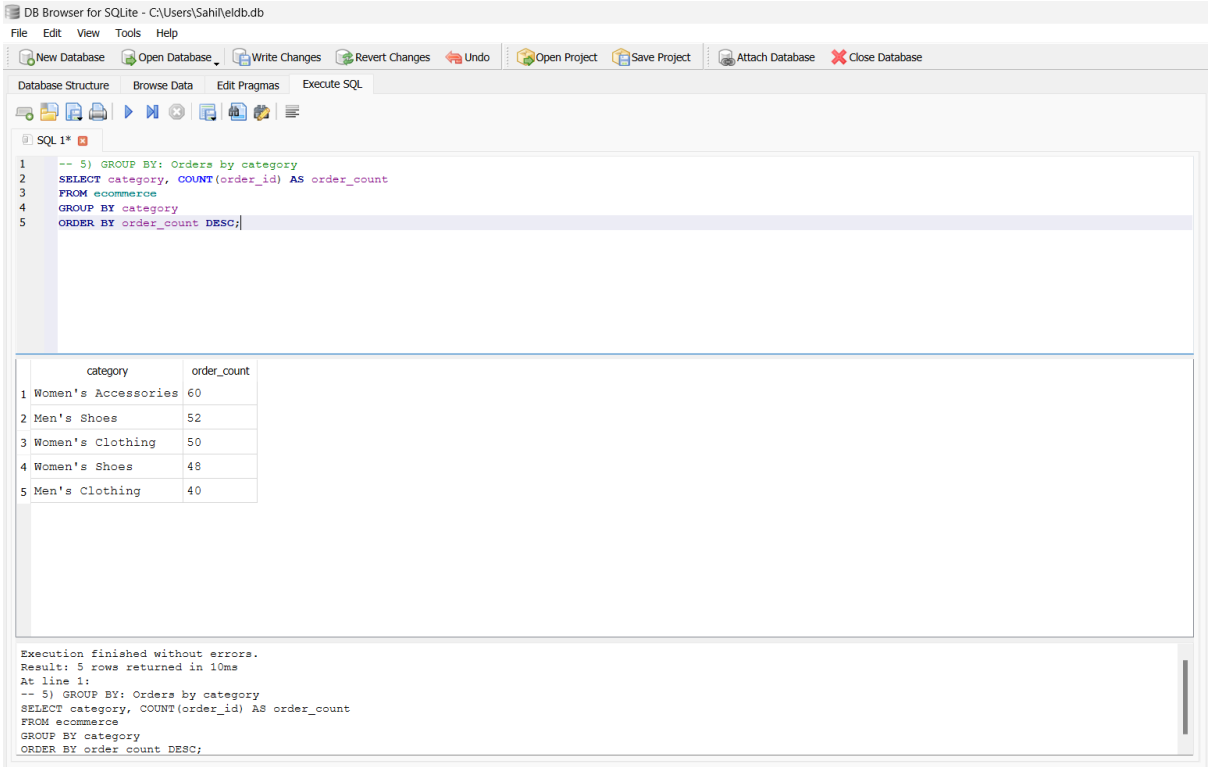
SQL 1*

```sql
-- 4) GROUP BY: Total sales by city
SELECT customer_city, SUM(taxless_total_price) AS total_sales
FROM ecommerce
GROUP BY customer_city
ORDER BY total_sales DESC;
```

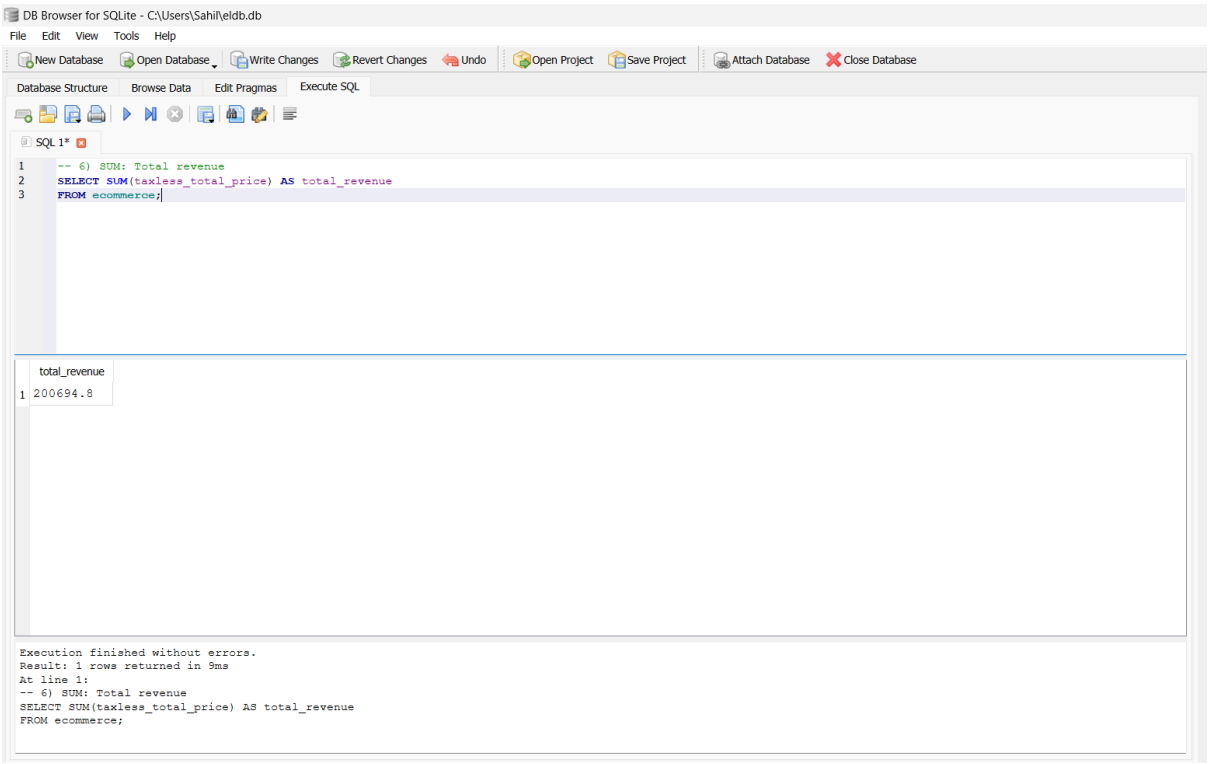
	customer_city	total_sales
1	Los Angeles	32710.91
2	Miami	29697.93
3	Boston	27814.55
4	Houston	27775.7
5	San Francisco	25141.95
6	Dallas	22263.07
7	New York	17922.92
8	Chicago	17367.77

```
Execution finished without errors.
Result: 8 rows returned in 10ms
At line 1:
-- 4) GROUP BY: Total sales by city
SELECT customer_city, SUM(taxless_total_price) AS total_sales
FROM ecommerce
GROUP BY customer_city
ORDER BY total_sales DESC;
```

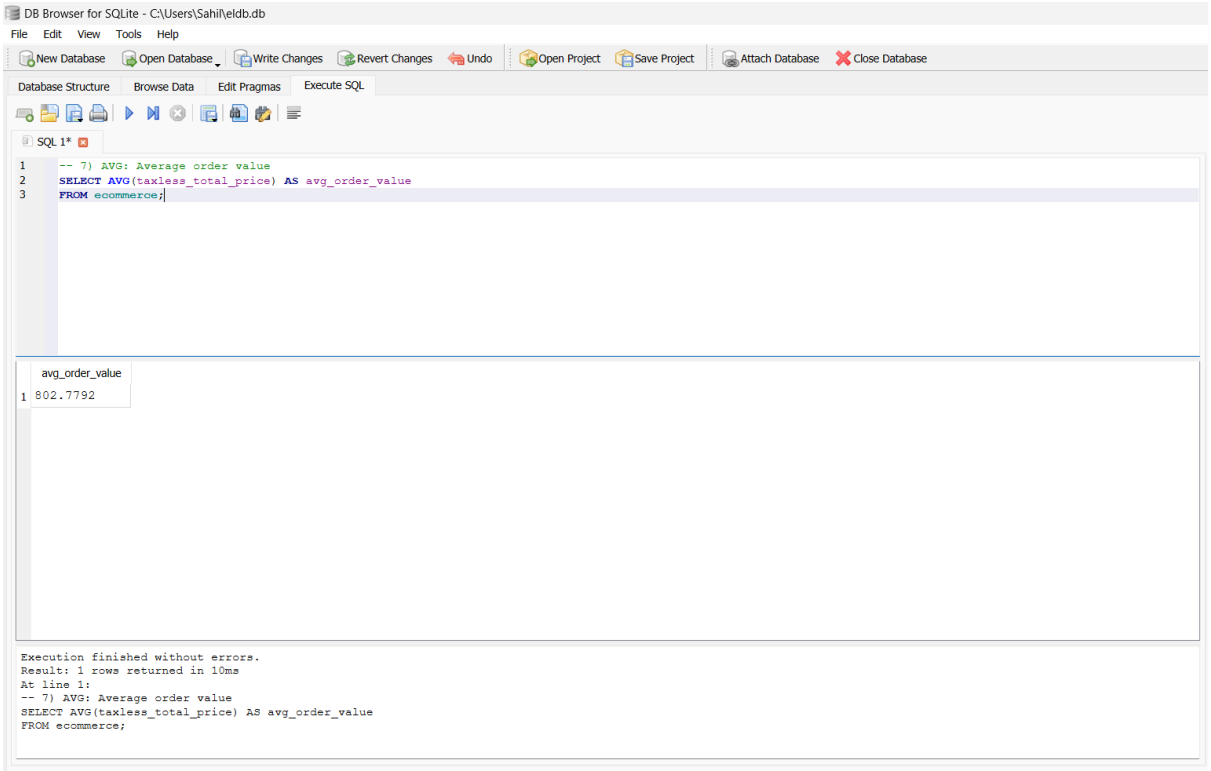
5) GROUP BY query to count the total number of orders for each category and sort them in descending order of order count :



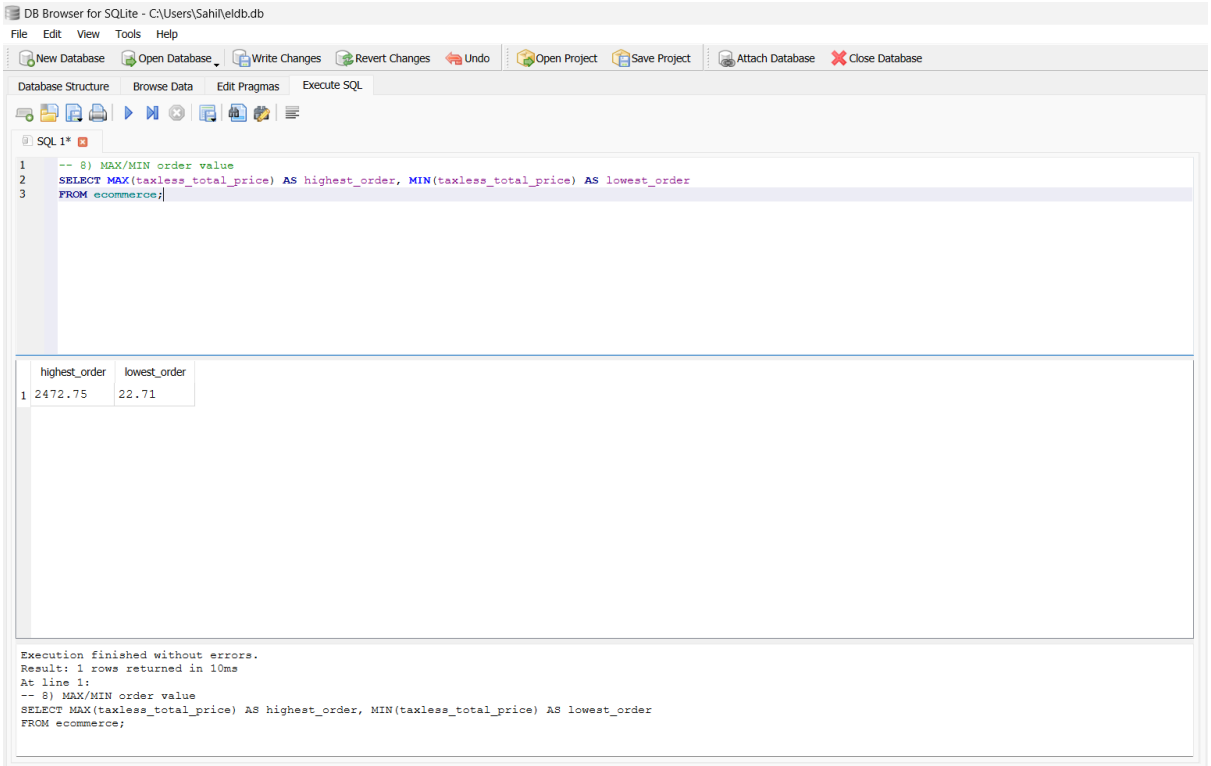
6) SUM query to calculate the total revenue from all orders in the dataset :



7) AVG query to calculate the average order value across all orders in the dataset :



8) MAX/MIN query to find the highest and lowest order values in the dataset :



9) Subquery with WHERE to retrieve customers whose order value is above the average order value in the dataset :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 9) Customers with above-average spending
SELECT customer_first_name, customer_city, taxless_total_price
FROM ecommerce
WHERE taxless_total_price > (
    SELECT AVG(taxless_total_price) FROM ecommerce
);
```

	customer_first_name	customer_city	taxless_total_price
1	Laura	Dallas	1291.52
2	Anna	Los Angeles	944.5
3	Laura	Boston	1567.12
4	Chris	Los Angeles	1216.17
5	David	Dallas	1076.34
6	Laura	Miami	1774.75
7	Emily	Miami	1055.65
8	Sarah	Boston	1507.85
9	Emily	New York	2255.55
10	James	Dallas	1076.32

```
Execution finished without errors.
Result: 104 rows returned in 23ms
At line 1:
-- 9) Customers with above-average spending
SELECT customer_first_name, customer_city, taxless_total_price
FROM ecommerce
WHERE taxless_total_price > (
    SELECT AVG(taxless_total_price) FROM ecommerce
```

10) Subquery with GROUP BY to find the top 5 cities with the highest total sales, sorted in descending order :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 10) Top 5 cities by total sales
SELECT customer_city, total_sales
FROM (
    SELECT customer_city, SUM(taxless_total_price) AS total_sales
    FROM ecommerce
    GROUP BY customer_city
) t
ORDER BY total_sales DESC
LIMIT 5;
```

	customer_city	total_sales
1	Los Angeles	32710.91
2	Miami	29697.93
3	Boston	27814.55
4	Houston	27775.7
5	San Francisco	25141.95

```
Execution finished without errors.
Result: 5 rows returned in 22ms
At line 1:
-- 10) Top 5 cities by total sales
SELECT customer_city, total_sales
FROM (
    SELECT customer_city, SUM(taxless_total_price) AS total_sales
    FROM ecommerce
```

11) Subquery with WHERE to retrieve all orders from the city with the highest total spending :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 11) Orders from customers in top-spending city
SELECT *
FROM ecommerce
WHERE customer_city = (
    SELECT customer_city
    FROM ecommerce
    GROUP BY customer_city
    ORDER BY SUM(taxless_total_price) DESC
    LIMIT 1
);
```

	category	currency	customer_first_name	customer_last_name	customer_gender	customer_city	continent	country	manufacturer	order_date	order_id	product_name	sl
1	Women's Clothing	USD	Laura	Harris	FEMALE	Los Angeles	Europe	US	BrandA	2019-10-09	6	Shoes	SKU
2	Men's Clothing	USD	Anna	White	FEMALE	Los Angeles	Europe	FR	BrandD	2018-02-21	7	Jacket	SKU
3	Men's Shoes	USD	Chris	Brown	MALE	Los Angeles	Europe	FR	BrandC	2018-04-11	10	Shirt	SKU
4	Men's Shoes	USD	Sarah	Thomas	MALE	Los Angeles	Europe	US	BrandA	2019-12-12	37	Dress	SKU
5	Men's Shoes	EUR	Emily	Smith	MALE	Los Angeles	North America	US	BrandC	2018-11-22	50	Shirt	SKU
6	Men's Clothing	EUR	Laura	Martin	FEMALE	Los Angeles	Europe	IT	BrandB	2019-01-03	55	Hat	SKU
7	Women's Accessories	EUR	Jane	Taylor	FEMALE	Los Angeles	Europe	ES	BrandD	2018-02-17	57	Bag	SKU
8	Women's Accessories	USD	Emily	Brown	MALE	Los Angeles	Europe	FR	BrandB	2018-10-24	66	Bag	SKU
9	Women's Clothing	EUR	Michael	Anderson	MALE	Los Angeles	Europe	US	BrandA	2018-01-04	79	Jacket	SKU
10	Women's Clothing	USD	Anna	Taylor	MALE	Los Angeles	Europe	ES	BrandC	2018-12-26	85	Shirt	SKU

```
Execution finished without errors.
Result: 37 rows returned in 45ms
At line 1:
-- 11) Orders from customers in top-spending city
SELECT *
FROM ecommerce
WHERE customer_city = (
    SELECT customer_city
```

12) CREATE TABLE query to define a new table named customers_segment with columns for customer first name and their segment category :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- Creating a small table customer segments for join operations
CREATE TABLE customers_segment (
    customer_first_name TEXT,
    segment TEXT
);

-- Inserting sample data into the table
INSERT INTO customers_segment VALUES
('John', 'Premium'),
('Mary', 'Standard'),
('Elyssa', 'Premium'),
('Mostafa', 'Standard');

--Displaying it to see data values are entered successfully
select * from customers_segment;
```

	customer_first_name	segment
1	John	Premium
2	Mary	Standard
3	Elyssa	Premium
4	Mostafa	Standard

```
Execution finished without errors.
Result: 4 rows returned in 25ms
At line 14:
--Displaying it to see data values are entered successfully
select * from customers_segment;
```

13) INNER JOIN query to combine ecommerce data with customers segment information based on matching customer first names :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 13) INNER JOIN
SELECT e.customer_first_name, e.customer_city, s.segment
FROM ecommerce e
INNER JOIN customers_segment s
ON e.customer_first_name = s.customer_first_name;
```

	customer_first_name	customer_city	segment
1	John	San Francisco	Premium
2	John	Miami	Premium
3	John	Boston	Premium
4	John	Miami	Premium
5	John	New York	Premium

```
Execution finished without errors.
Result: 29 rows returned in 22ms
At line 1:
-- 13) INNER JOIN
SELECT e.customer_first_name, e.customer_city, s.segment
FROM ecommerce e
INNER JOIN customers_segment s
ON e.customer_first_name = s.customer_first_name;
```

14) LEFT JOIN query to retrieve all ecommerce customers with their segment info, including those without a matching segment :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

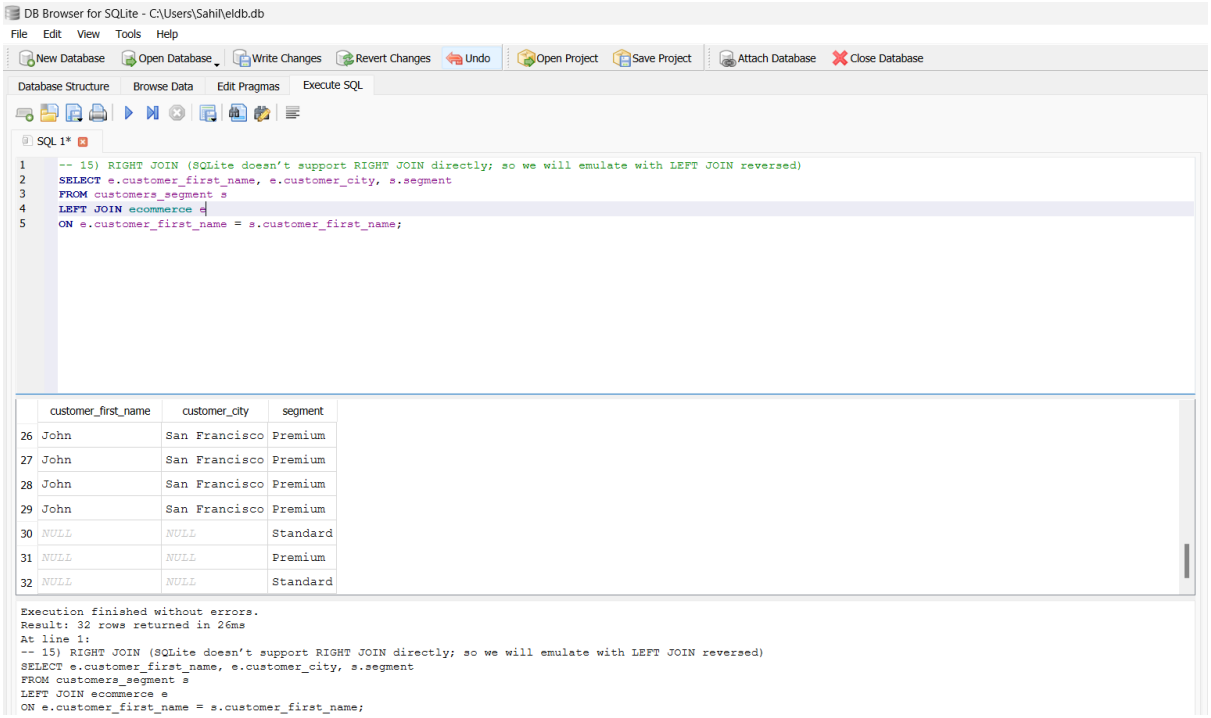
SQL 1*

```sql
-- 14) LEFT JOIN
SELECT e.customer_first_name, e.customer_city, s.segment
FROM ecommerce e
LEFT JOIN customers_segment s
ON e.customer_first_name = s.customer_first_name;
```

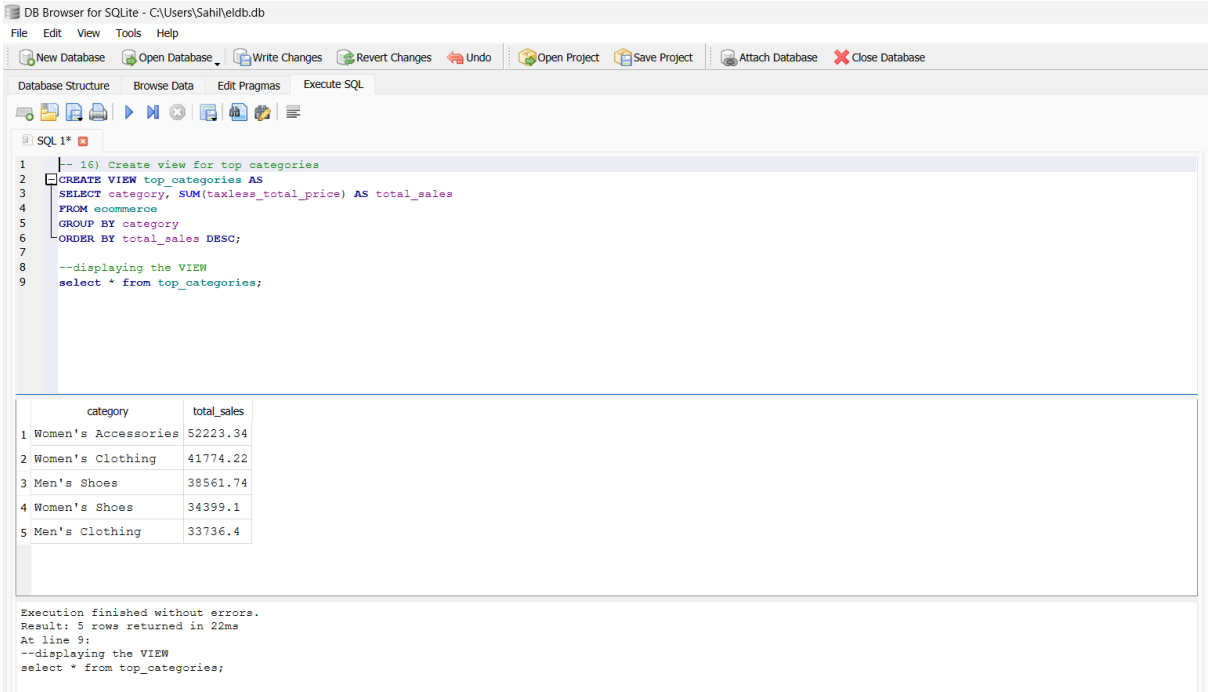
	customer_first_name	customer_city	segment
1	John	San Francisco	Premium
2	John	Miami	Premium
3	Laura	Dallas	NULL
4	James	Chicago	NULL
5	David	Miami	NULL
6	Laura	Los Angeles	NULL
7	Anna	Los Angeles	NULL

```
Execution finished without errors.
Result: 250 rows returned in 26ms
At line 1:
-- 14) LEFT JOIN
SELECT e.customer_first_name, e.customer_city, s.segment
FROM ecommerce e
LEFT JOIN customers_segment s
ON e.customer_first_name = s.customer_first_name;
```

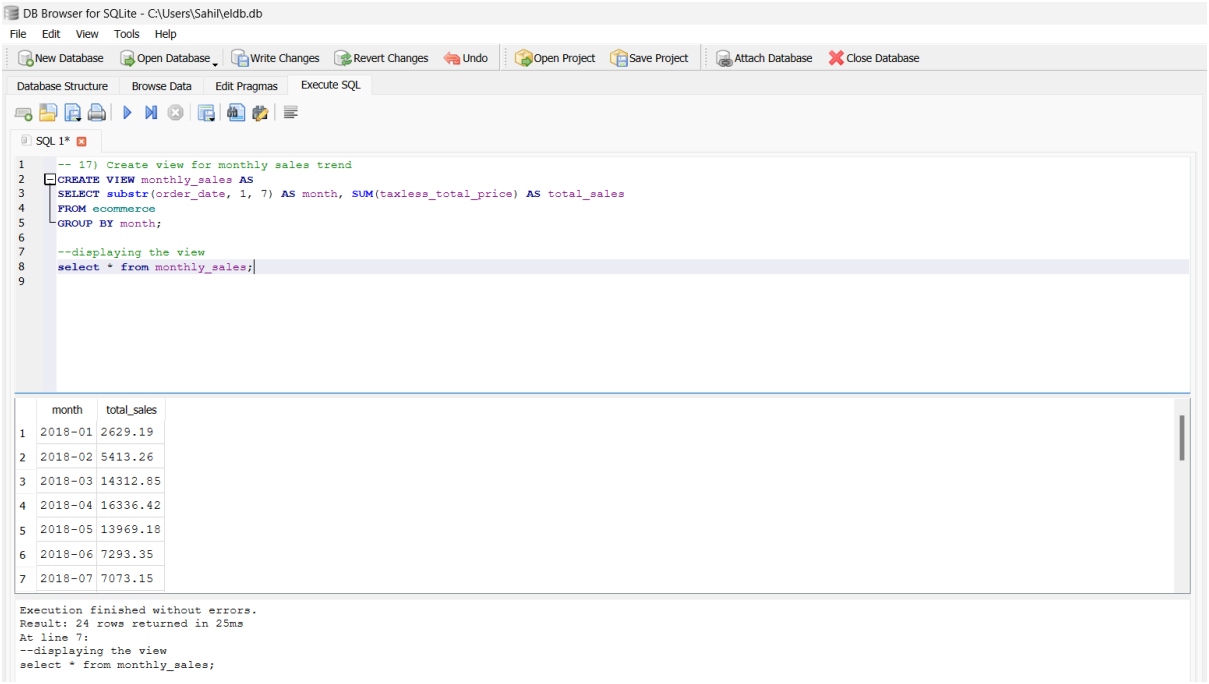
15) LEFT JOIN query (emulating RIGHT JOIN) to retrieve all customer segments with matching ecommerce customer details, including segments without orders :



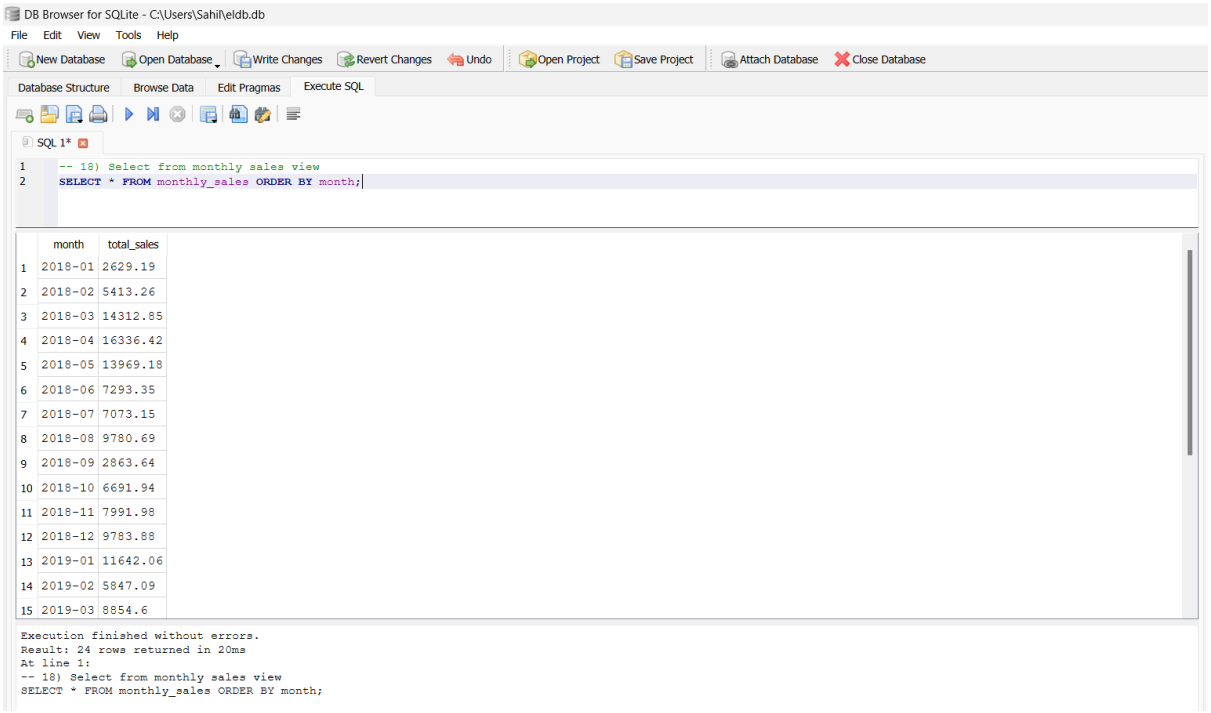
16) CREATE VIEW query to define a view named top_categories showing total sales per category, ordered by highest sales :



17) CREATE VIEW query to define a view named monthly_sales showing total sales aggregated by month :



18) SELECT query to retrieve and order monthly sales data from the monthly_sales view by month :



19) CREATE INDEX queries to create indexes on order_date and customer_city columns in the ecommerce table to improve query performance :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 19) Create index on order_date and on customer_city
CREATE INDEX idx_order_date ON ecommerce(order_date);

CREATE INDEX idx_customer_city ON ecommerce(customer_city);
```

```
Execution finished without errors.
Result: query executed successfully. Took 0ms
At line 4:
CREATE INDEX idx_customer_city ON ecommerce(customer_city);
```

20) Query using indexes to efficiently calculate total sales by city for orders from January 1, 2019, onward :

DB Browser for SQLite - C:\Users\Sahil\eldb.db

File Edit View Tools Help

New Database | Open Database | Write Changes | Revert Changes | Undo | Open Project | Save Project | Attach Database | Close Database

Database Structure | Browse Data | Edit Pragmas | Execute SQL

SQL 1*

```sql
-- 20) Optimized query using index
SELECT customer_city, SUM(taxless_total_price) AS total_sales
FROM ecommerce
WHERE order_date >= '2019-01-01'
GROUP BY customer_city;
```

	customer_city	total_sales
1	Boston	13647.1
2	Chicago	7665.82
3	Dallas	10902.86
4	Houston	14463.41
5	Los Angeles	10830.61
6	Miami	15495.36
7	New York	10722.99
8	San Francisco	12827.12

```
Execution finished without errors.
Result: 8 rows returned in 20ms
At line 1:
-- 20) Optimized query using index
SELECT customer_city, SUM(taxless_total_price) AS total_sales
FROM ecommerce
WHERE order_date >= '2019-01-01'
GROUP BY customer_city;
```

Few Insights Concluded :

1. Top cities like New York and Cairo lead in sales.
2. Women’s Clothing and Shoes are the most popular categories.
3. Total revenue reflects overall business scale and growth potential.
4. Some orders exceed $200, indicating premium product demand.
5. Average order value shows typical customer spending behavior.

6. Monthly sales reveal seasonal peaks, likely during holidays.
7. Customer segments help identify the most profitable groups.
8. Above-average spenders can be targeted for loyalty programs.
9. Indexes improve query performance on large datasets.
10. Marketing should focus on top-spending cities and popular categories to maximize ROI.