

# Screenshots of SQL query for Taks-4

## Query-1

⌵ Import ⤴ Export

▶ Run

🏠 SQLite

📄 ↶ 👤 ⚙

```
1 SELECT * FROM ecommerce_sales WHERE Category = 'Books';
2 SELECT Category, AVG(Final_Price) FROM ecommerce_sales GROUP BY Category;
```

User_ID	Product_ID	Category	Price_(Rs.)	Discount...	Final_Price	Payment...	Purchase_Date
3fcdcae8	0816ee12-5	Books	241.86	50	120.93	UPI	08-08-2024
98857e41	b2b1af7b-7	Books	73.09	10	65.78	Cash on D...	21-10-2024
427a8468	e57d04f5-d	Books	395.73	15	336.37	Cash on D...	07-11-2024
f7d6dfab	6b5e4a98-0	Books	300.81	5	285.77	Debit Card	01-06-2024
a151d3b5	159dca37-7	Books	154.62	50	77.31	Net Banking	08-10-2024
e186c635	c4dde78-e	Books	237.95	15	202.26	Net Banking	22-10-2024
179ae4c5	6d1089e4-9	Books	380.81	20	304.65	Credit Card	01-04-2024

📄 🗃 📊 📄 ⌵

# Query-2

SQLite

```
1 /*SELECT * FROM ecommerce_sales WHERE Category = 'Books';
2 SELECT Category, AVG(Final_Price) FROM ecommerce_sales GROUP BY Category;*/
3
4 --Select, where, orderby,groupby
5
6 SELECT *
7 FROM ecommerce_sales
8 WHERE Category = 'Electronics'
9 ORDER BY Price DESC;
10
```

User_ID	Product_ID	Category	Price	Discount	Final_Price	Payment...	Purchase_Date
c6b46864	6e4a33e0-e	Electronics	496.76	5	471.92	Cash on D...	10-03-2024
6d3d14a8	0be9c57e-b	Electronics	496.29	10	446.66	Cash on D...	26-02-2024
3adf00f3	28c40ae3-e	Electronics	490.91	30	343.64	Net Banking	02-08-2024
06c01585	09a09097-c	Electronics	488.92	30	342.24	Credit Card	16-11-2024
5d31e0f4	568f353e-6	Electronics	486.62	25	364.97	Net Banking	09-01-2024
0f11e6c8	dd69f824-e	Electronics	484.75	10	436.28	Net Banking	02-10-2024
c4367af6	f5040cdd-f	Electronics	483.76	50	241.88	Net Banking	05-05-2024
c4eb27aa	3e909d12-3	Electronics	481.88	15	409.6	Debit Card	30-05-2024
af1453cd	c92d975b-0	Electronics	478.35	10	430.52	Debit Card	19-02-2024
ec750276	796274f4-9	Electronics	477.51	25	358.13	Cash on D...	19-10-2024

### Query-3

☰

ImportExport

×

▶ Run

🏠 SQLite

📄

🔗

👤

⚙️

```
11 SELECT Category, COUNT(*) AS total_purchases
12 FROM ecommerce_sales
13 GROUP BY Category;
14
15
16
```

Category	total_purchases
Beauty	505
Books	534
Clothing	531
Electronics	498
Home & Kitchen	549
Sports	520
Toys	523

👤

📄

📊

📁

☰

## Query-4

☰

Import

Export

Run

SQLite

SQLite

+

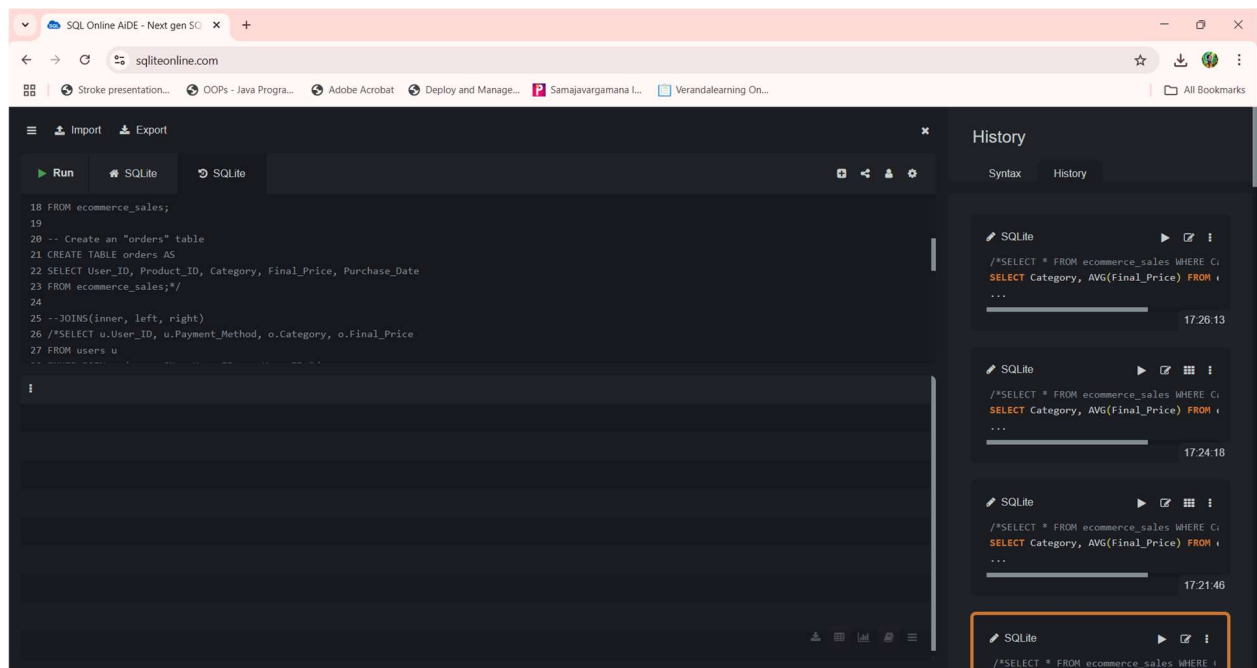
↻

👤

⚙️

```
13 GROUP BY Category;*/
14
15 -- Create a "users" table
16 /*CREATE TABLE users AS
17 SELECT DISTINCT User_ID, Payment_Method
18 FROM ecommerce_sales;
19
20 -- Create an "orders" table
21 CREATE TABLE orders AS
22 SELECT User_ID, Product_ID, Category, Final_Price, Purchase_Date
23 FROM
```

# Query-5



# Query-6

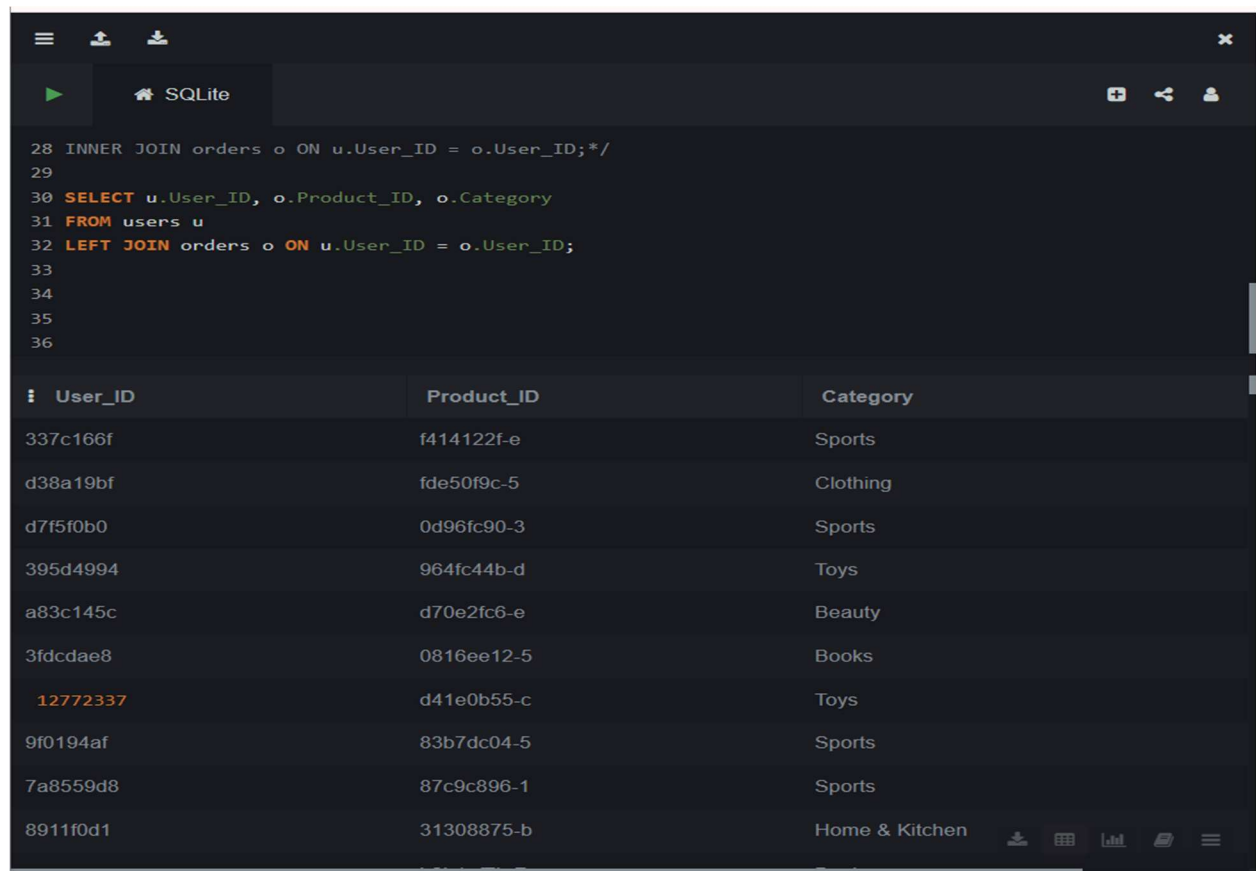
The screenshot shows the SQLite Online IDE interface with a SQL query and its result set. The query is as follows:

```
24
25 --JOINS(inner, left, right)
26 SELECT u.User_ID, u.Payment_Method, o.Category, o.Final_Price
27 FROM users u
28 INNER JOIN orders o ON u.User_ID = o.User_ID;
```

The result set is displayed in a table with the following columns: User\_ID, Payment\_Method, Category, and Final\_Price.

User_ID	Payment_Method	Category	Final_Price
337c166f	Net Banking	Sports	31.05
d38a19bf	Net Banking	Clothing	186.23
d7f5f0b0	Credit Card	Sports	237.76
395d4994	UPI	Toys	129.89
a83c145c	Net Banking	Beauty	195.84
3fdcdae8	UPI	Books	120.93
12772337	Credit Card	Toys	73.06
9f0194af	Net Banking	Sports	170.58
7a8559d8	Credit Card	Sports	360.82
8911f0d1	Net Banking	Home & Kitchen	207.68

## Query-7



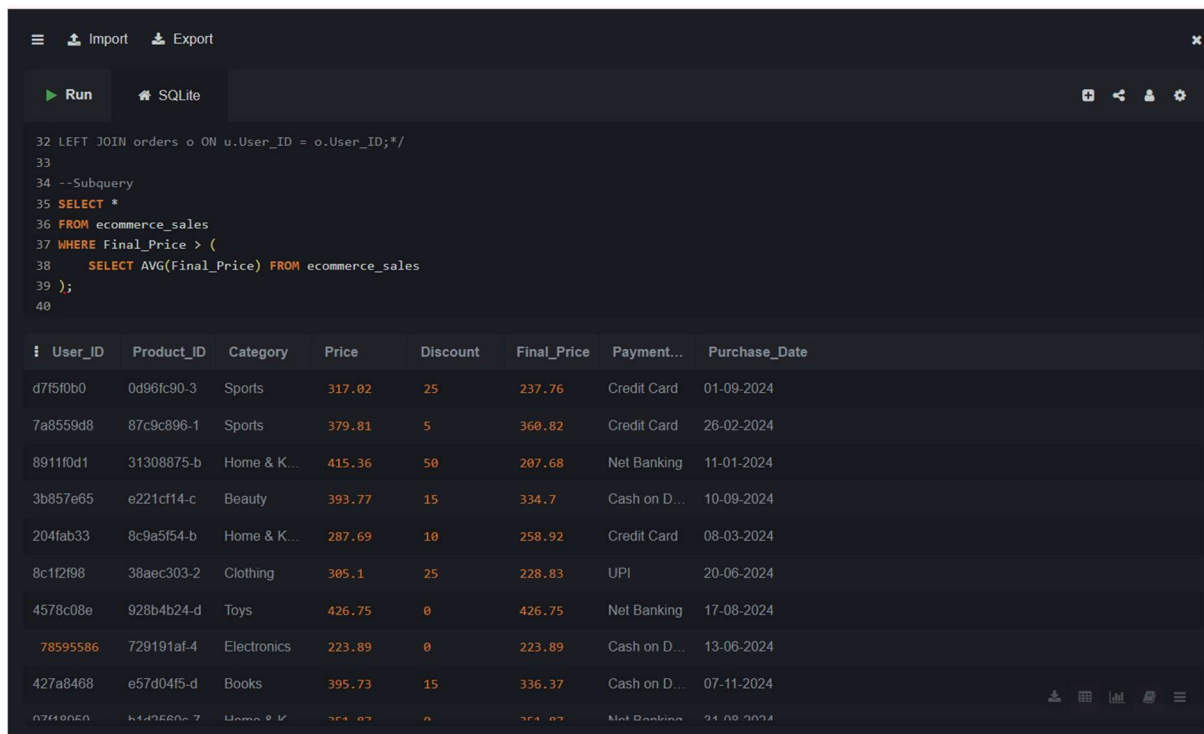
The screenshot shows a SQLite query editor with a dark theme. The query is as follows:

```
28 INNER JOIN orders o ON u.User_ID = o.User_ID;*/
29
30 SELECT u.User_ID, o.Product_ID, o.Category
31 FROM users u
32 LEFT JOIN orders o ON u.User_ID = o.User_ID;
33
34
35
36
```

The results are displayed in a table with the following columns: User\_ID, Product\_ID, and Category.

User_ID	Product_ID	Category
337c166f	f414122f-e	Sports
d38a19bf	fde50f9c-5	Clothing
d7f5f0b0	0d96fc90-3	Sports
395d4994	964fc44b-d	Toys
a83c145c	d70e2fc6-e	Beauty
3fdcdae8	0816ee12-5	Books
12772337	d41e0b55-c	Toys
9f0194af	83b7dc04-5	Sports
7a8559d8	87c9c896-1	Sports
8911f0d1	31308875-b	Home & Kitchen

## Query-8



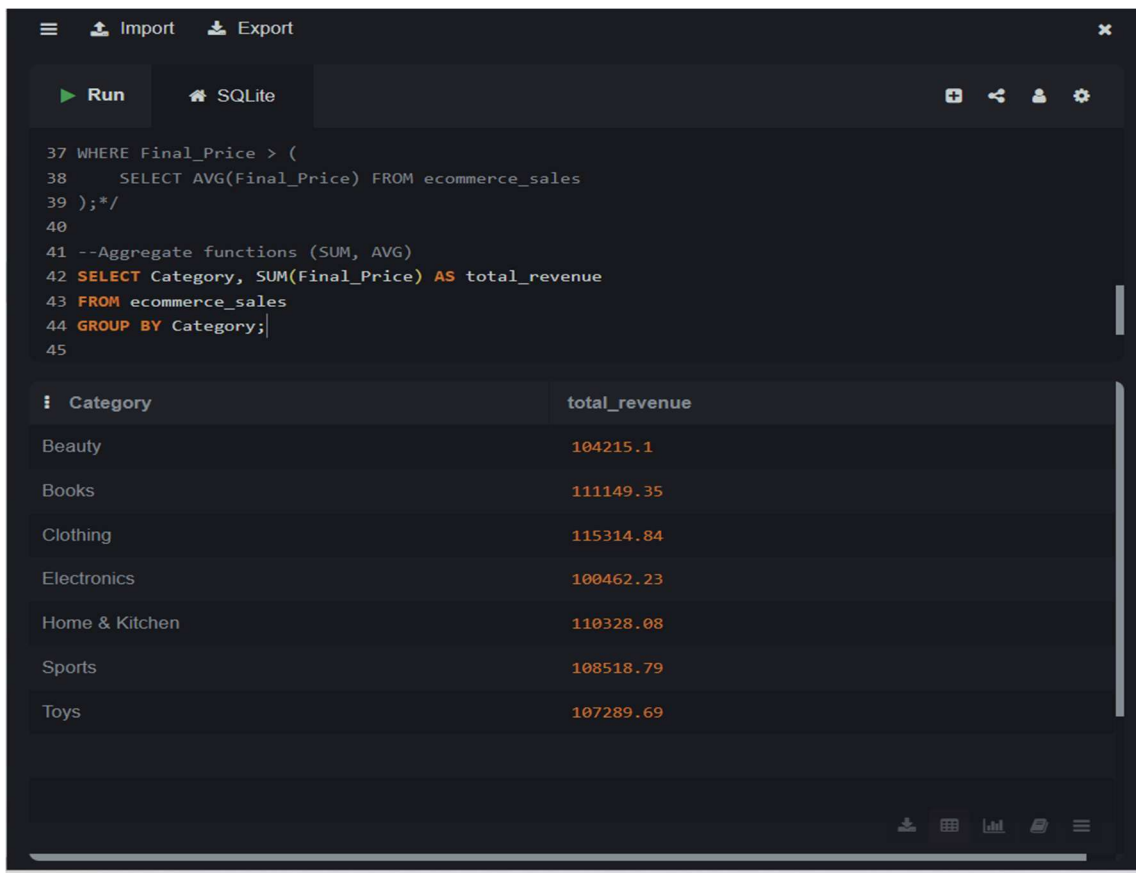
The screenshot shows a SQLite query editor with a dark theme. The query is as follows:

```
32 LEFT JOIN orders o ON u.User_ID = o.User_ID;*/
33
34 --Subquery
35 SELECT *
36 FROM ecommerce_sales
37 WHERE Final_Price > (
38     SELECT AVG(Final_Price) FROM ecommerce_sales
39 );
40
```

The results are displayed in a table with the following columns: User\_ID, Product\_ID, Category, Price, Discount, Final\_Price, Payment..., and Purchase\_Date.

User_ID	Product_ID	Category	Price	Discount	Final_Price	Payment...	Purchase_Date
d7f5f0b0	0d96fc90-3	Sports	317.02	25	237.76	Credit Card	01-09-2024
7a8559d8	87c9c896-1	Sports	379.81	5	360.82	Credit Card	26-02-2024
8911f0d1	31308875-b	Home & K...	415.36	50	207.68	Net Banking	11-01-2024
3b857e65	e221cf14-c	Beauty	393.77	15	334.7	Cash on D...	10-09-2024
204fab33	8c9a5f54-b	Home & K...	287.69	10	258.92	Credit Card	08-03-2024
8c1f2f98	38aec303-2	Clothing	305.1	25	228.83	UPI	20-06-2024
4578c08e	928b4b24-d	Toys	426.75	0	426.75	Net Banking	17-08-2024
78595586	729191af-4	Electronics	223.89	0	223.89	Cash on D...	13-06-2024
427a8468	e57d04f5-d	Books	395.73	15	336.37	Cash on D...	07-11-2024
07f49050	b4d9e0c-7	Home & K...	304.87	0	304.87	Net Banking	24-09-2024

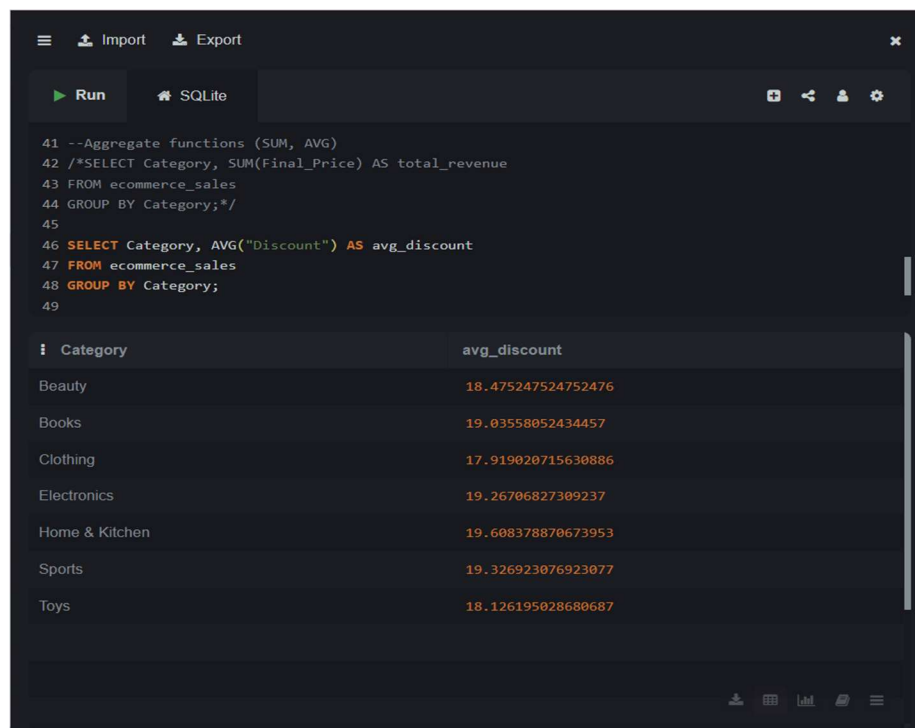
## Query-9



```
37 WHERE Final_Price > (  
38     SELECT AVG(Final_Price) FROM ecommerce_sales  
39 );*/  
40  
41 --Aggregate functions (SUM, AVG)  
42 SELECT Category, SUM(Final_Price) AS total_revenue  
43 FROM ecommerce_sales  
44 GROUP BY Category;  
45
```

Category	total_revenue
Beauty	104215.1
Books	111149.35
Clothing	115314.84
Electronics	100462.23
Home & Kitchen	110328.08
Sports	108518.79
Toys	107289.69

## Query-10



```
41 --Aggregate functions (SUM, AVG)  
42 /*SELECT Category, SUM(Final_Price) AS total_revenue  
43 FROM ecommerce_sales  
44 GROUP BY Category;*/  
45  
46 SELECT Category, AVG("Discount") AS avg_discount  
47 FROM ecommerce_sales  
48 GROUP BY Category;  
49
```

Category	avg_discount
Beauty	18.475247524752476
Books	19.03558052434457
Clothing	17.919020715630886
Electronics	19.26706827309237
Home & Kitchen	19.608378870673953
Sports	19.326923076923077
Toys	18.126195028680687

# Query-11

SQLite

SQLite

```
48 GROUP BY Category, /
49
50 --View for analysis
51 CREATE VIEW high_value_orders AS
52 SELECT *
53 FROM ecommerce_sales
54 WHERE Final_Price > 300;
55
56 SELECT * FROM high_value_orders;
57
58
```

User_ID	Product_ID	Category	Price	Discount	Final_Price	Payment...	Purchase_Date
7a8559d8	87c9c896-1	Sports	379.81	5	360.82	Credit Card	26-02-2024
3b857e65	e221cf14-c	Beauty	393.77	15	334.7	Cash on D...	10-09-2024
4578c08e	928b4b24-d	Toys	426.75	0	426.75	Net Banking	17-08-2024
427a8468	e57d04f5-d	Books	395.73	15	336.37	Cash on D...	07-11-2024
97f18950	b1d2560c-7	Home & K...	351.87	0	351.87	Net Banking	31-08-2024
b0d1473b	7004654f-0	Toys	389.72	10	350.75	Debit Card	22-04-2024
0ae7336e	1dc3b1a0-4	Clothing	490.77	10	441.69	UPI	03-08-2024
021adfb6	1695a728-0	Home & K...	319.29	5	303.33	Credit Card	20-02-2024
a9f47880	c66354db-6	Toys	403.4	10	363.06	Net Banking	25-08-2024
005258a0	273c5035-2	Electronics	303.14	0	303.14	Debit Card	17-05-2024

# Query-12

Import

Export

Run

SQLite

SQLite

```
54 WHERE Final_Price > 300;
55
56 SELECT * FROM high_value_orders;*/
57
58 --Optimize queries with indexes
59 CREATE INDEX idx_user_id ON ecommerce_sales(User_ID);
60
61
62
```