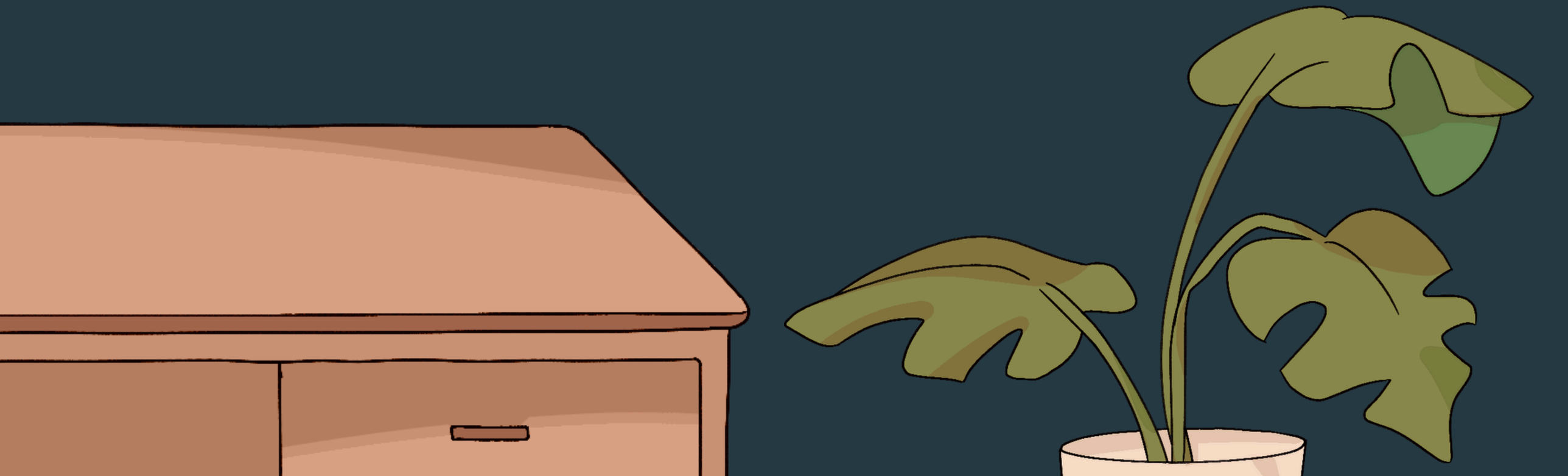
BOOKS STORE ANALYSIS

Presented by: Hemang Taori

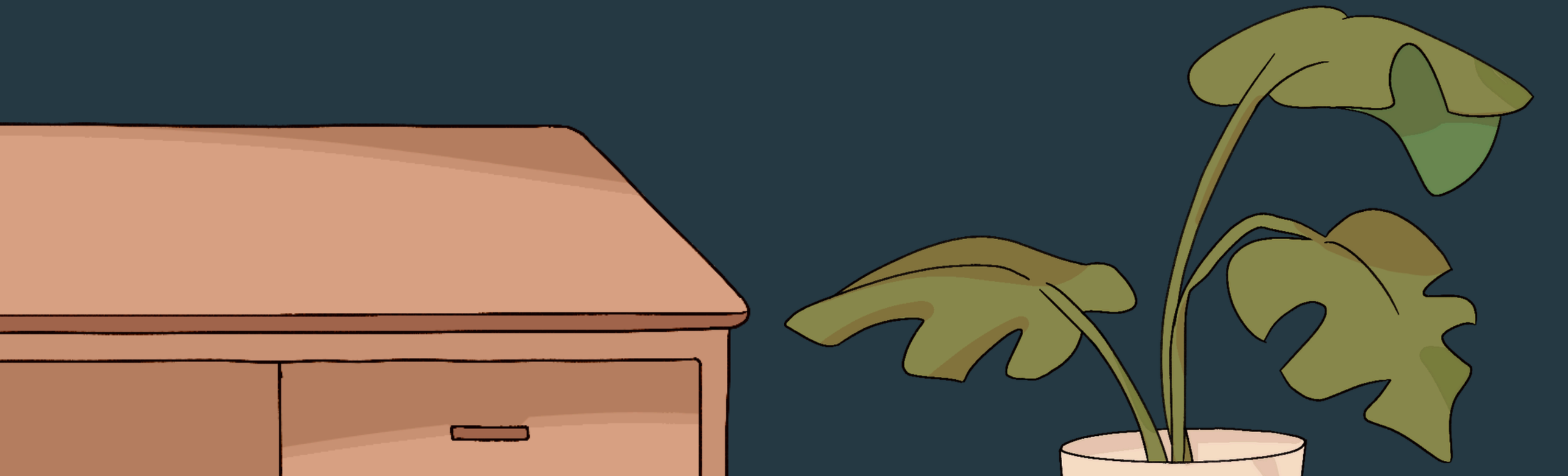
1. FIND THE TOTAL NUMBER OF BOOKS SOLD FOR EACH GENRE.

```sql
select books.genre, coalesce(sum(orders.quantity),0) as quantity
from books left join orders
on books.book_id = orders.book_id
group by books.genre order by quantity desc;
```



2.FIND THE AVERAGE PRICE OF BOOKS IN THE 'FANTASY' GENRE.

```sql
select sum(books.price)/count(books.book_id) from books
where books.genre = 'Fantasy';
```

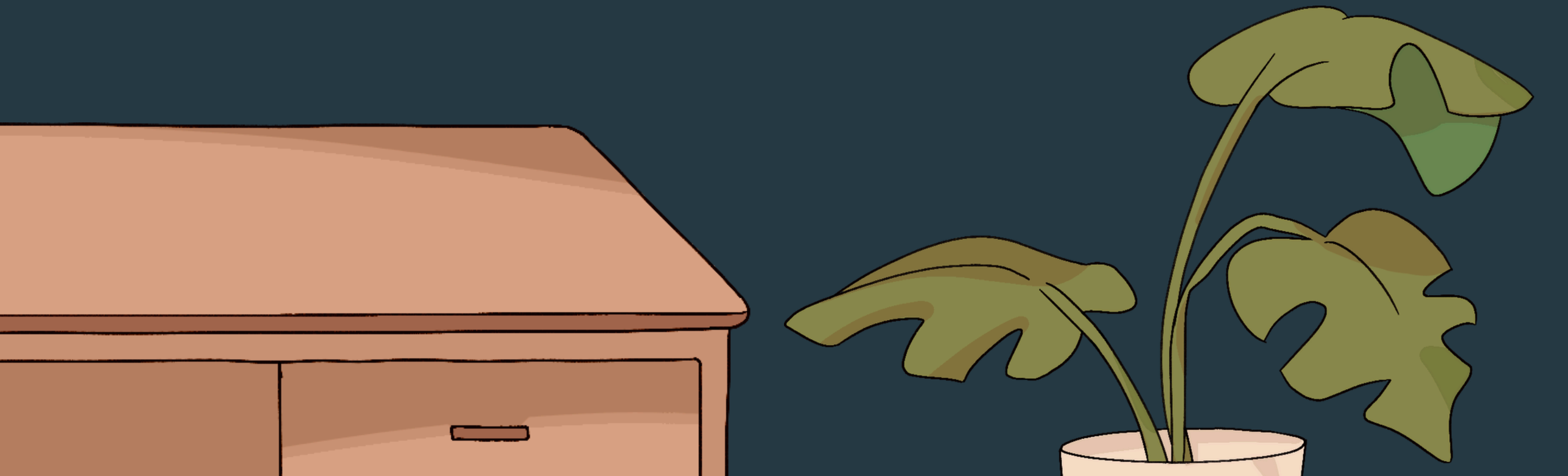


3. LIST THE CUSTOMERS WHO HAVE PLACED AT LEAST 2 ORDERS

```sql
select customers.name, sum(orders.quantity) as sns
from customers join orders
on customers.customer_id = orders.customer_id
group by customers.name
having sns>=2
order by sns desc;
```

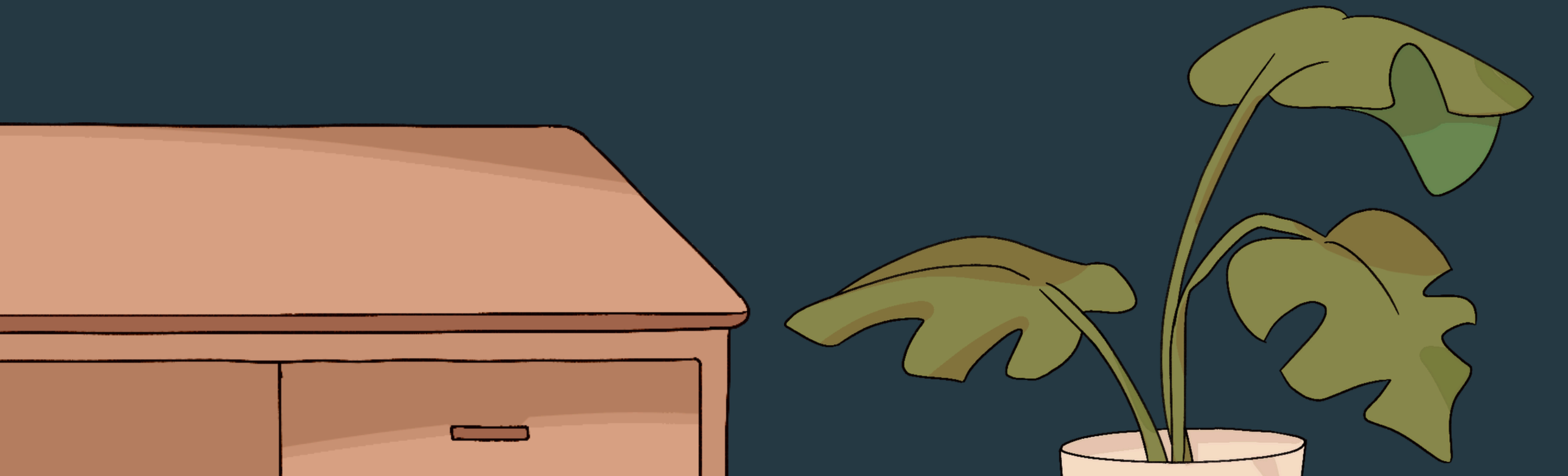
4. FIND THE MOST FREQUENTLY ORDERED BOOK

```sql
select books.title, count(orders.order_id) as sns
from books join orders
on books.book_id = orders.book_id
group by books.title order by sns desc limit 1;
```



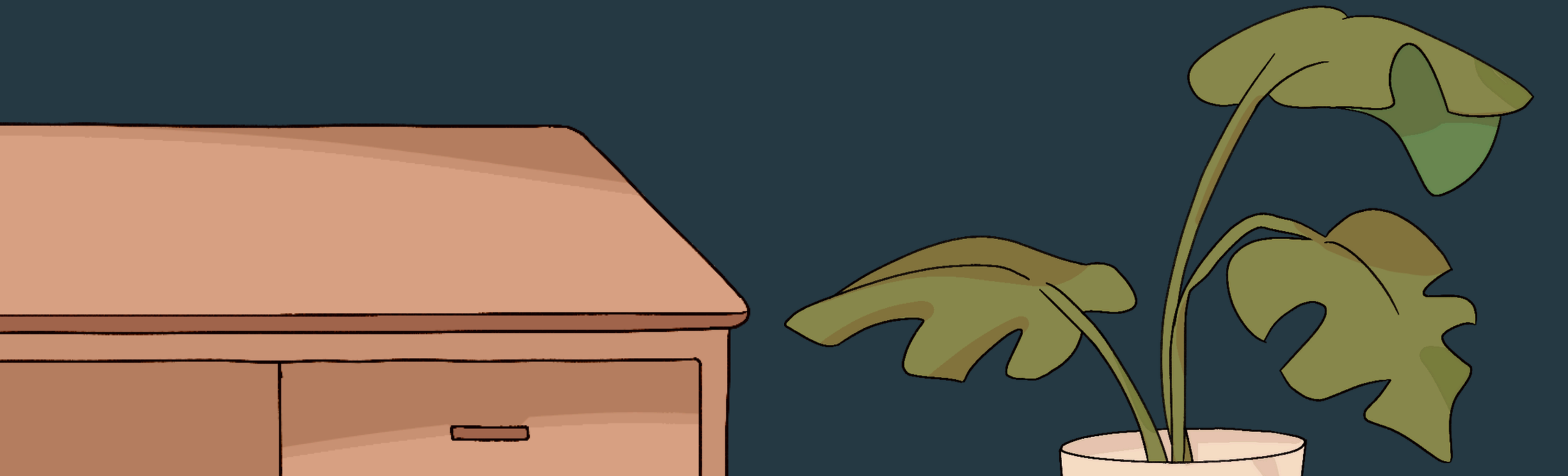
5. SHOW THE TOP 3 MOST EXPENSIVE BOOKS OF FANTASY GENRE

```sql
select title, price from books
where genre= "Fantasy"
order by price desc limit 3;
```



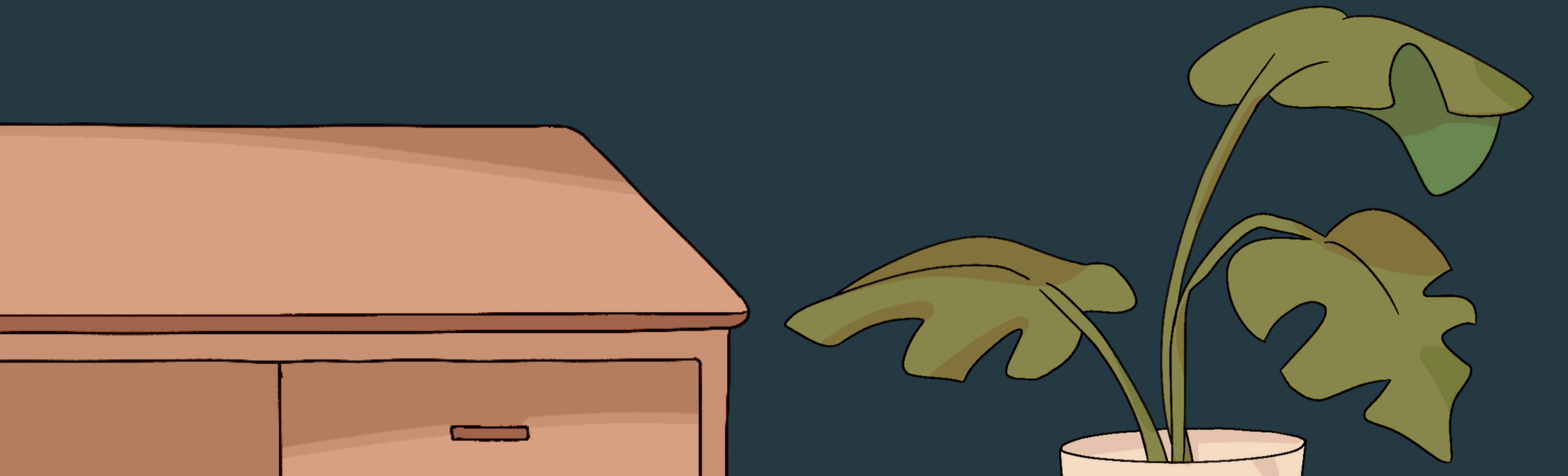
6. RETRIVE THE TOTAL QUANTITY OF BOOKS SOLD BY EACH AUTHOR.

```sql
select books.author, sum(orders.quantity) as sum
from books join orders
on books.book_id = orders.book_id
group by books.author order by sum desc;
```



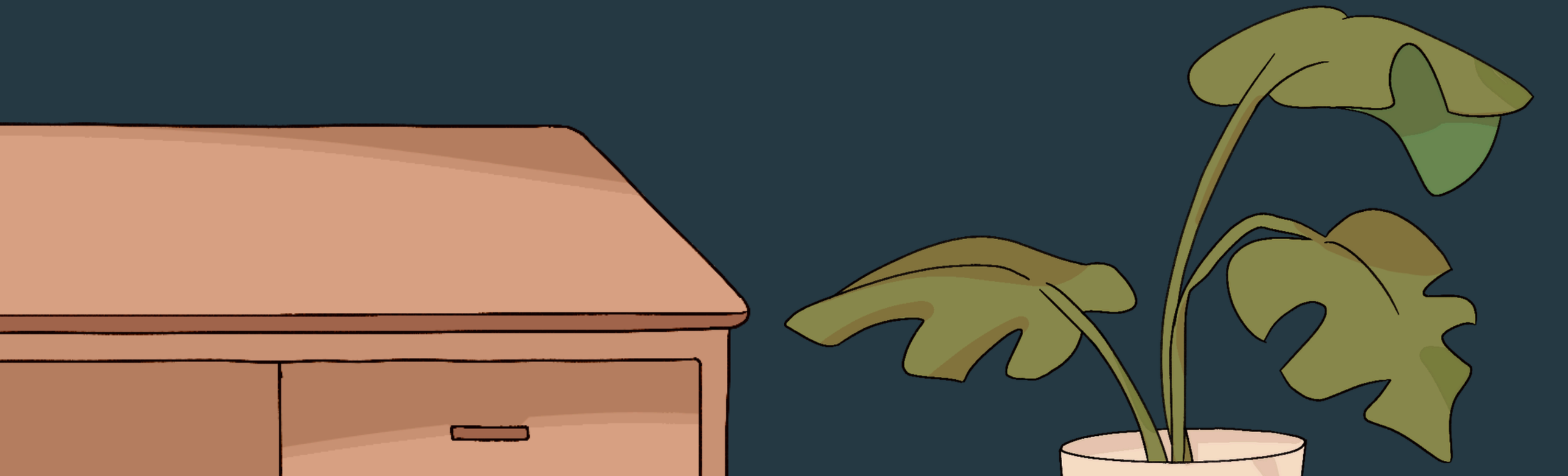
7. LIST THE CITIES WHERE CUSTOMERS WHO SPENT OVER $30 ARE LOCATED.

```sql
select customers.name, customers.city, sum(orders.total_amount) as sns
from customers join orders
on customers.customer_id = orders.customer_id
group by customers.name, customers.city
having sns >=30
order by sns desc;
```



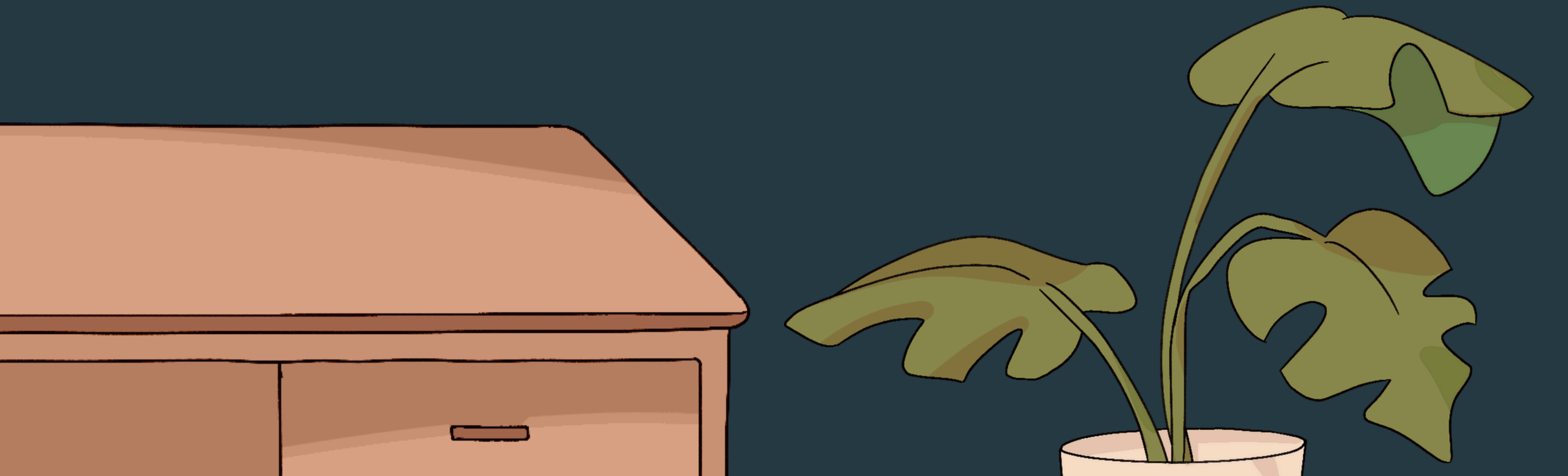
8. FIND THE CUSTOMER WHO HASS SPEND MOST ON ORDERS

```sql
select customers.name, sum(orders.total_amount) as sns
from customers join orders
on customers.customer_id = orders.customer_id
group by customers.name
having sns >=30
order by sns desc limit 1;
```



9. CALCULATE THE STOCK REMAINING AFTER FULFILLING ALL THE ORDERS. WHERE THE STOCK IS RUNNING SHORT AS COMPARED TO ORDERS

```sql
# calculate the stock remaining after fulfilling all the orders. where the stock is running short as compared to orders
select books.title, books.stock, coalesce(sum(orders.quantity),0) as total_ordered, books.stock - coalesce(sum(orders.quantity),0) as stock_remaining
from books left join orders
on books.book_id = orders.book_id
group by books.title, books.stock
having stock_remaining <0
order by stock_remaining asc;
```



CONCLUSION

- In summary, this SQL project focused on answering key business questions related to customer behavior, sales performance, and product popularity by analyzing data from three interconnected Excel files: Customers, Orders, and Books. By leveraging SQL functionalities such as multiple joins, filtering conditions, grouping, sorting, and aggregation functions, we were able to extract meaningful insights from raw data. These insights helped the business owner understand which books were performing well, which customers were most active, which genres were popular in different regions, and how revenue was distributed. All questions were answered effectively using SQL, transforming data into strategic business intelligence

THANK YOU

Don't hesitate to ask any questions!