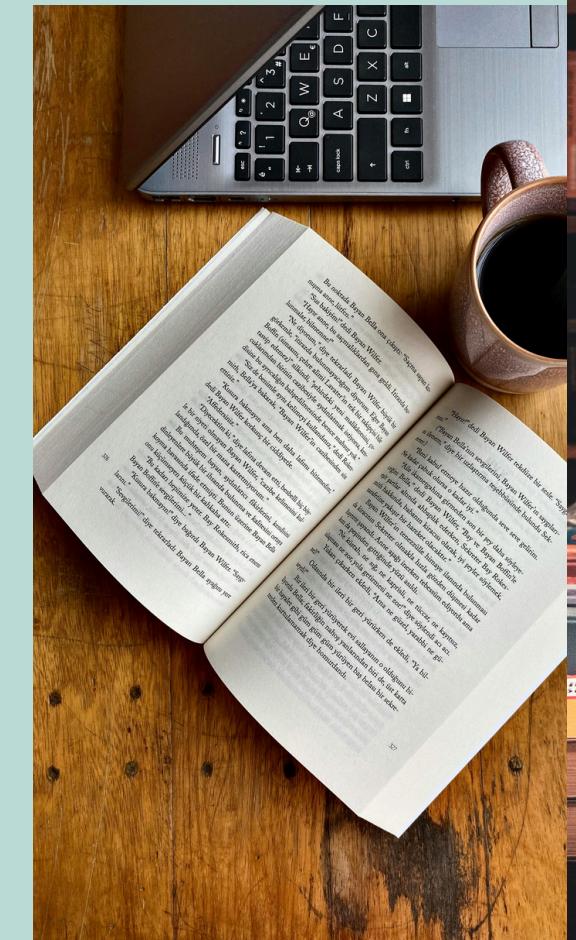


SQL PROJECT ON ONLINE BOOK STORE SALES

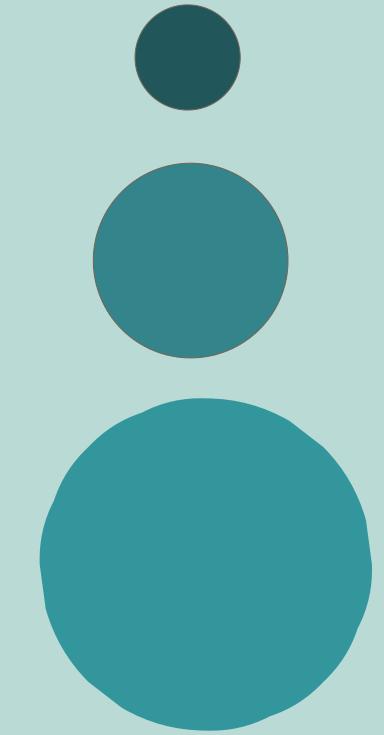
HELLO

My Name is SADIQUA QUADIR and in this project, I have utilised SQL Queries to solve questions related to online book sales.



Retrieve all books in the "Fiction" genre.

```
SELECT * FROM Books  
WHERE Genre="Fiction";
```



	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	4	Customizable 24hour product	Christopher Andrews	Fiction	2020	43.52	8
	22	Multi-layered optimizing migration	Wesley Escobar	Fiction	1908	39.23	78
	28	Expanded analyzing portal	Lisa Coffey	Fiction	1941	37.51	79
	29	Quality-focused multi-tasking challenge	Katrina Underwood	Fiction	1905	31.12	100
	31	Implemented encompassing conglomeration	Melissa Taylor	Fiction	2010	21.23	44
	39	Optimized national process improvement	Megan Goodwin	Fiction	1978	10.99	42
	40	Adaptive didactic interface	Natalie Gonzalez	Fiction	1923	25.97	94
	47	Reverse-engineered directional conglomeration	John Christian	Fiction	2006	20.37	90
	62	Re-contextualized real-time strategy	Nicole Lynch	Fiction	1953	26.34	23
	63	Revised business database	Paulina White	Fiction	1900	22.22	55

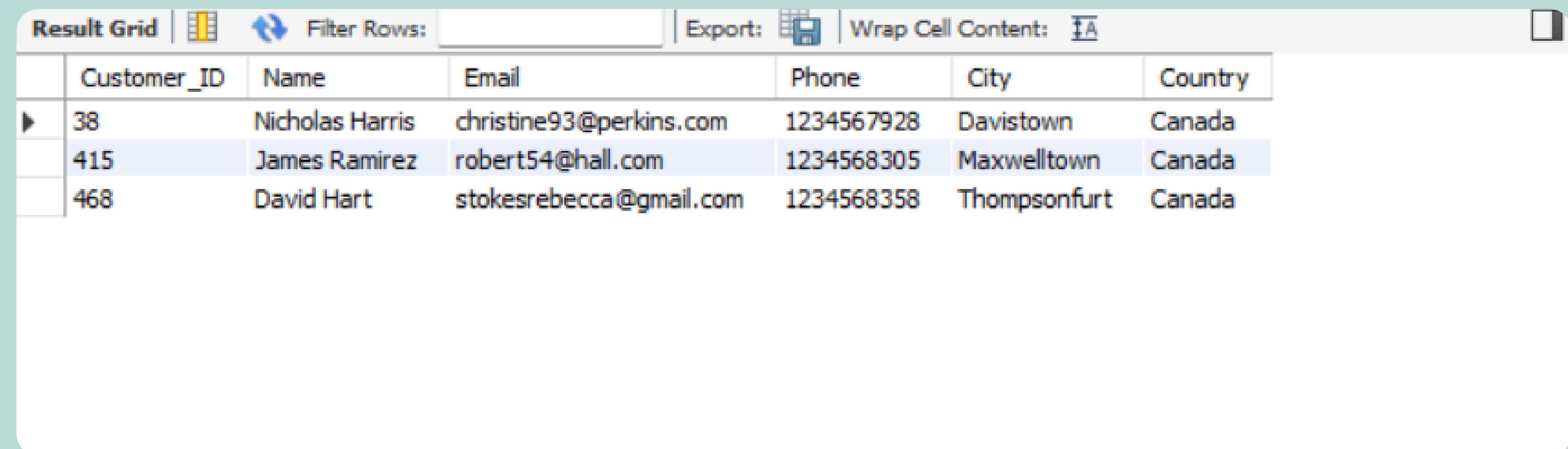
Find books published after the year 1950.

```
SELECT * FROM Books  
WHERE Published_Year>1950;
```

	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	2	Persevering reciprocal knowledge user	Mario Moore	Fantasy	1971	35.8	19
	4	Customizable 24hour product	Christopher Andrews	Fiction	2020	43.52	8
	5	Adaptive 5thgeneration encoding	Juan Miller	Fantasy	1956	10.95	16
	6	Advanced encompassing implementation	Bryan Morgan	Biography	1985	6.56	2
	8	Persistent local encoding	Troy Cox	Science Fiction	2019	48.99	84
	9	Optimized interactive challenge	Colin Buckley	Fantasy	1987	14.33	70
	10	Ergonomic national hub	Samantha Ruiz	Mystery	2015	24.63	25
	11	Secured zero tolerance time-frame	Denise Barnes	Fantasy	1998	35.95	10
	12	Selected national.....	Pauline Smith	New Fiction	2020	27.42	62

List all customers from Canada.

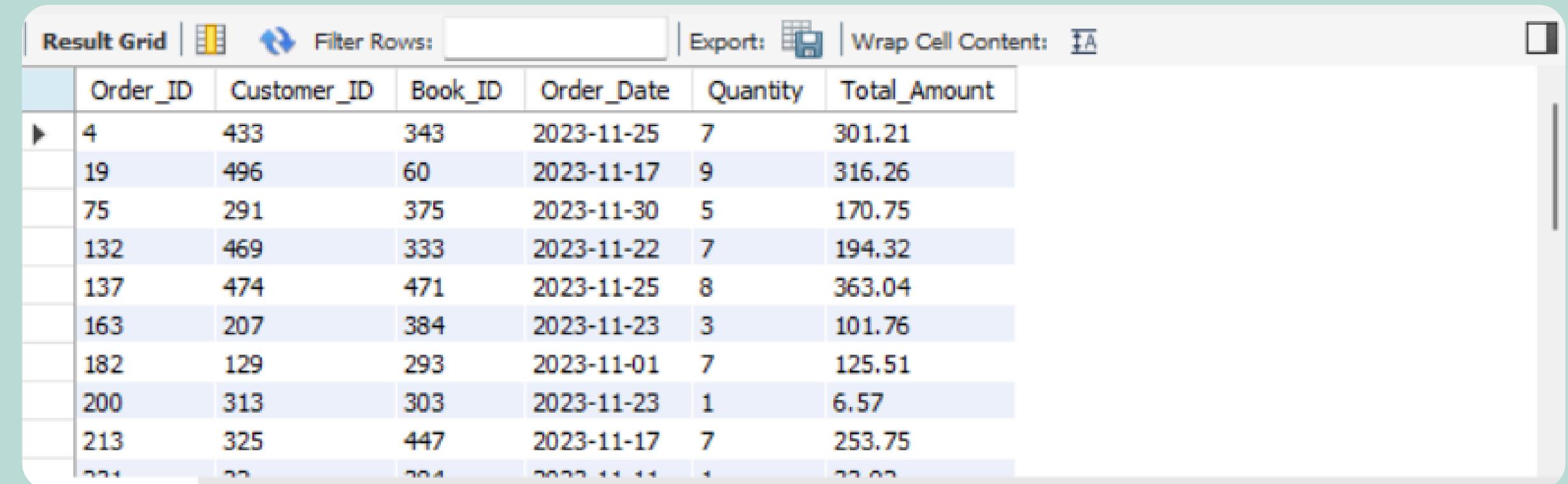
```
SELECT * FROM Customers  
WHERE Country = "Canada";
```



	Customer_ID	Name	Email	Phone	City	Country
▶	38	Nicholas Harris	christine93@perkins.com	1234567928	Davistown	Canada
	415	James Ramirez	robert54@hall.com	1234568305	Maxwelltown	Canada
	468	David Hart	stokesrebecca@gmail.com	1234568358	Thompsonfurt	Canada

Show orders placed in November 2023.

```
SELECT * FROM Orders  
WHERE Order_Date BETWEEN '2023-11-01' AND '2023-11-30';
```

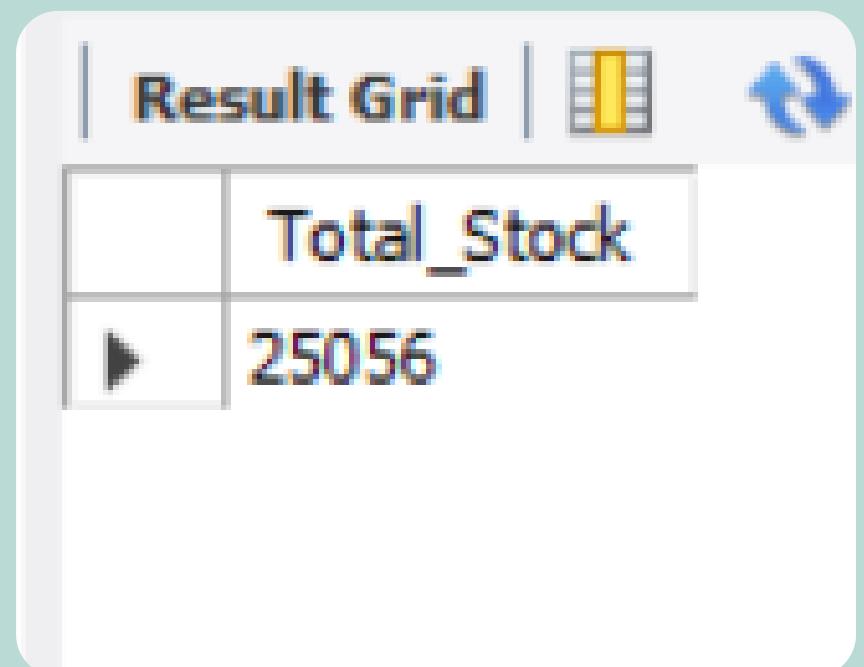


A screenshot of a database result grid titled "Result Grid". The grid displays a list of orders placed in November 2023. The columns are labeled: Order_ID, Customer_ID, Book_ID, Order_Date, Quantity, and Total_Amount. The data shows various purchases, such as order 4 for customer 433 on 2023-11-25 with a total amount of 301.21, and order 19 for customer 496 on 2023-11-17 with a total amount of 316.26. The grid has a header row and several data rows below it.

	Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
▶	4	433	343	2023-11-25	7	301.21
	19	496	60	2023-11-17	9	316.26
	75	291	375	2023-11-30	5	170.75
	132	469	333	2023-11-22	7	194.32
	137	474	471	2023-11-25	8	363.04
	163	207	384	2023-11-23	3	101.76
	182	129	293	2023-11-01	7	125.51
	200	313	303	2023-11-23	1	6.57
	213	325	447	2023-11-17	7	253.75
...

Retrieve the total stock of books available.

```
SELECT SUM(stock) AS Total_Stock  
FROM Books;
```

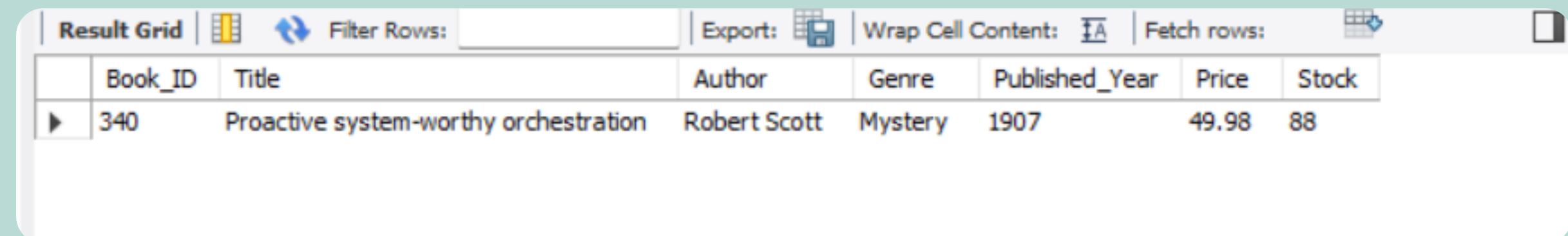


The screenshot shows a MySQL Workbench result grid. The title bar says "Result Grid". The grid has one row and two columns. The first column is labeled "Total_Stock" and contains the value "25056". There are navigation icons for previous and next results at the top right of the grid area.

Total_Stock	25056
-------------	-------

Find the details of the most expensive book.

```
SELECT * FROM Books  
ORDER BY Price DESC LIMIT 1;
```



The screenshot shows a database result grid titled "Result Grid". The grid has a header row with columns: Book_ID, Title, Author, Genre, Published_Year, Price, and Stock. A single row of data is displayed below the header:

	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	340	Proactive system-worthy orchestration	Robert Scott	Mystery	1907	49.98	88

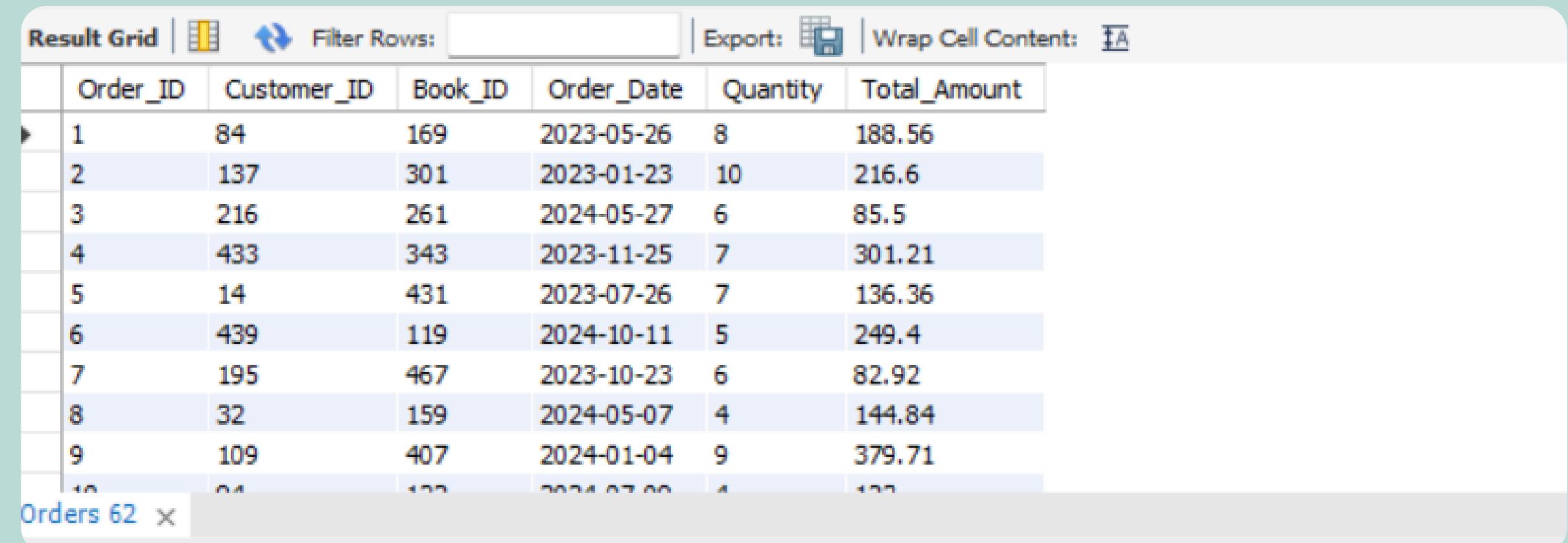
Show all customers who ordered more than 1 quantity of a book.

```
SELECT * FROM Orders  
WHERE Quantity>1;
```

	Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
▶	1	84	169	2023-05-26	8	188.56
	2	137	301	2023-01-23	10	216.6
	3	216	261	2024-05-27	6	85.5
	4	433	343	2023-11-25	7	301.21
	5	14	431	2023-07-26	7	136.36
	6	439	119	2024-10-11	5	249.4
	7	195	467	2023-10-23	6	82.92
	8	32	159	2024-05-07	4	144.84
	9	109	407	2024-01-04	9	379.71
	10	24	122	2024-07-09	4	122

Retrieve all orders where the total amount exceeds \$20.

```
SELECT * FROM Orders  
WHERE Total_Amount>20;
```



The screenshot shows a database query results grid titled "Orders 62". The grid has columns: Order_ID, Customer_ID, Book_ID, Order_Date, Quantity, and Total_Amount. The data is as follows:

	Order_ID	Customer_ID	Book_ID	Order_Date	Quantity	Total_Amount
1	84	169	2023-05-26	8	188.56	
2	137	301	2023-01-23	10	216.6	
3	216	261	2024-05-27	6	85.5	
4	433	343	2023-11-25	7	301.21	
5	14	431	2023-07-26	7	136.36	
6	439	119	2024-10-11	5	249.4	
7	195	467	2023-10-23	6	82.92	
8	32	159	2024-05-07	4	144.84	
9	109	407	2024-01-04	9	379.71	
10	62	122	2024-07-09	4	122	

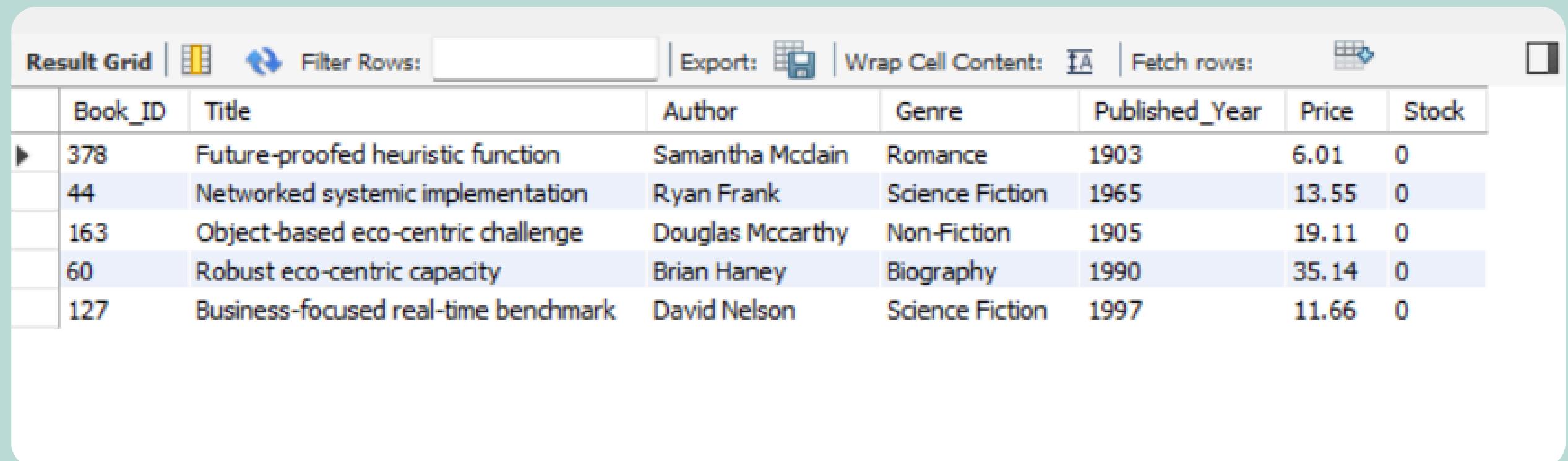
List all the genres available in the Books table.

```
SELECT DISTINCT Genre  
FROM Books;
```

Result Grid	
	Genre
▶	Biography
	Fantasy
	Non-Fiction
	Fiction
	Romance
	Science Fiction
	Mystery

Find the book with the lowest stock.

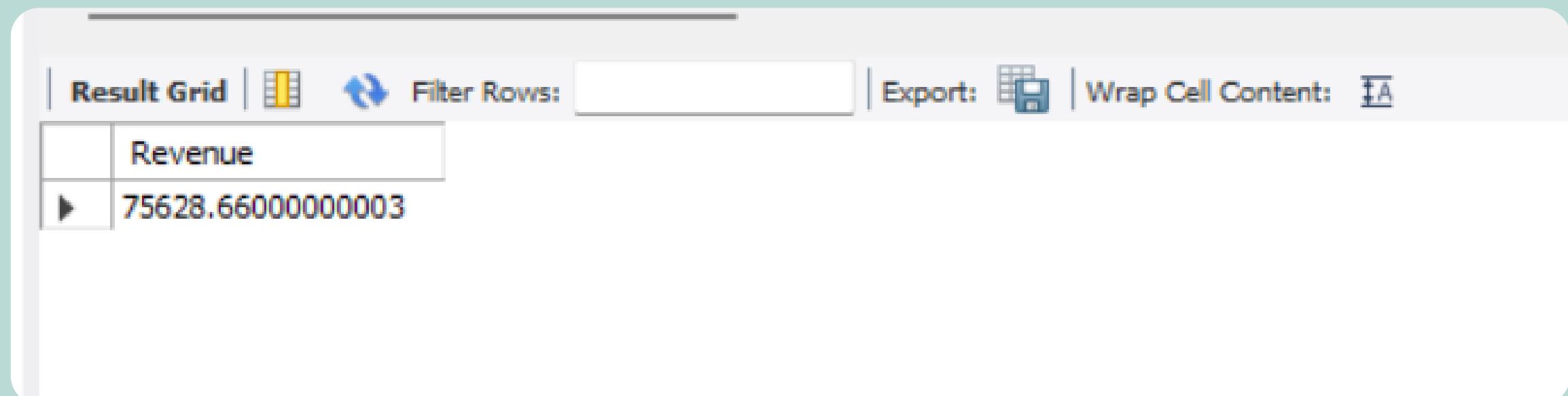
```
SELECT * FROM Books  
ORDER BY Stock ASC LIMIT 5;
```



	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	378	Future-proofed heuristic function	Samantha Mcdain	Romance	1903	6.01	0
	44	Networked systemic implementation	Ryan Frank	Science Fiction	1965	13.55	0
	163	Object-based eco-centric challenge	Douglas McCarthy	Non-Fiction	1905	19.11	0
	60	Robust eco-centric capacity	Brian Haney	Biography	1990	35.14	0
	127	Business-focused real-time benchmark	David Nelson	Science Fiction	1997	11.66	0

Calculate the total revenue generated from all orders.

```
SELECT SUM(Total_Amount) AS Revenue  
FROM Orders;
```

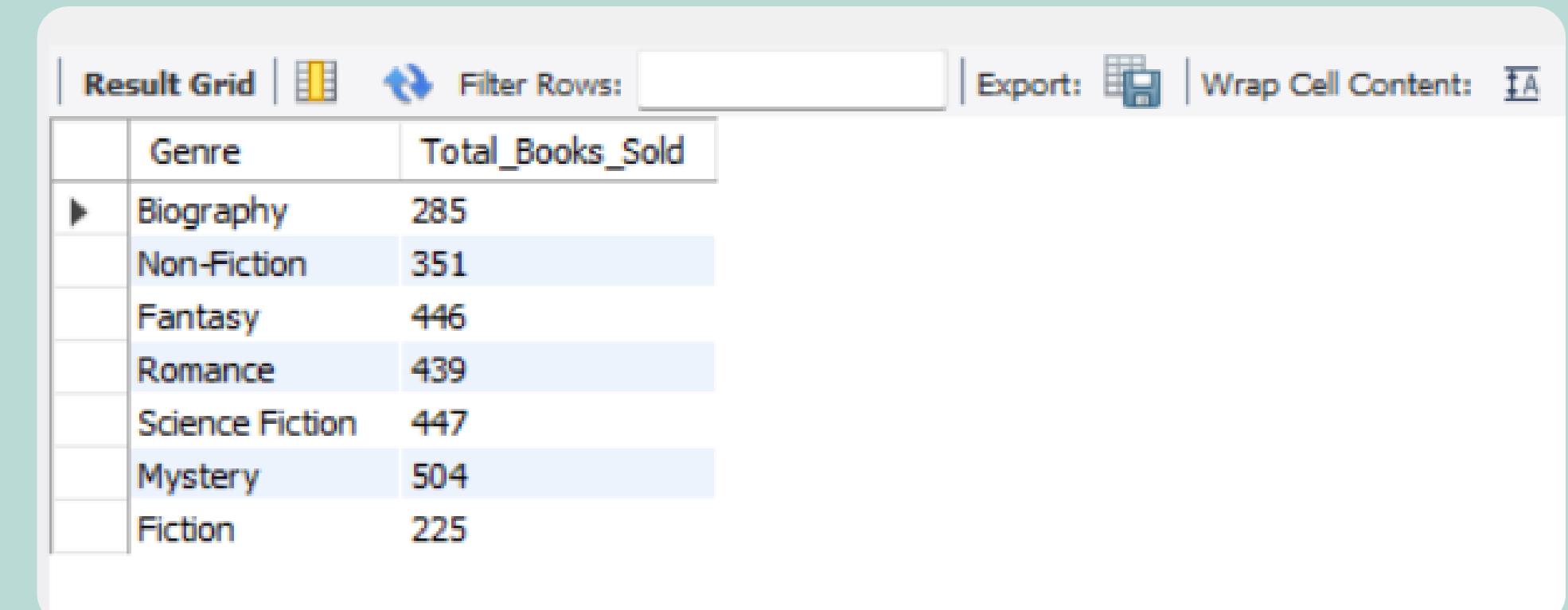


A screenshot of a database query results grid. The grid has a header row with two columns: one empty and one labeled "Revenue". Below the header is a single data row with a right-pointing arrow icon and the value "75628.66000000003". The grid is set against a light gray background with a white border. At the top of the grid, there is a toolbar with several icons: "Result Grid" (grid icon), "Filter Rows" (refresh icon), "Export" (grid icon), and "Wrap Cell Content" (text icon). There is also a search bar with the placeholder text "Filter Rows:".

	Revenue
▶	75628.66000000003

Retrieve the total number of books sold for each genre.

```
SELECT * FROM Orders;
SELECT B.Genre , SUM(O.Quantity) AS
Total_Books_Sold
FROM Orders O
JOIN Books B ON O.Book_ID = B.Book_ID
GROUP BY B.Genre;
```

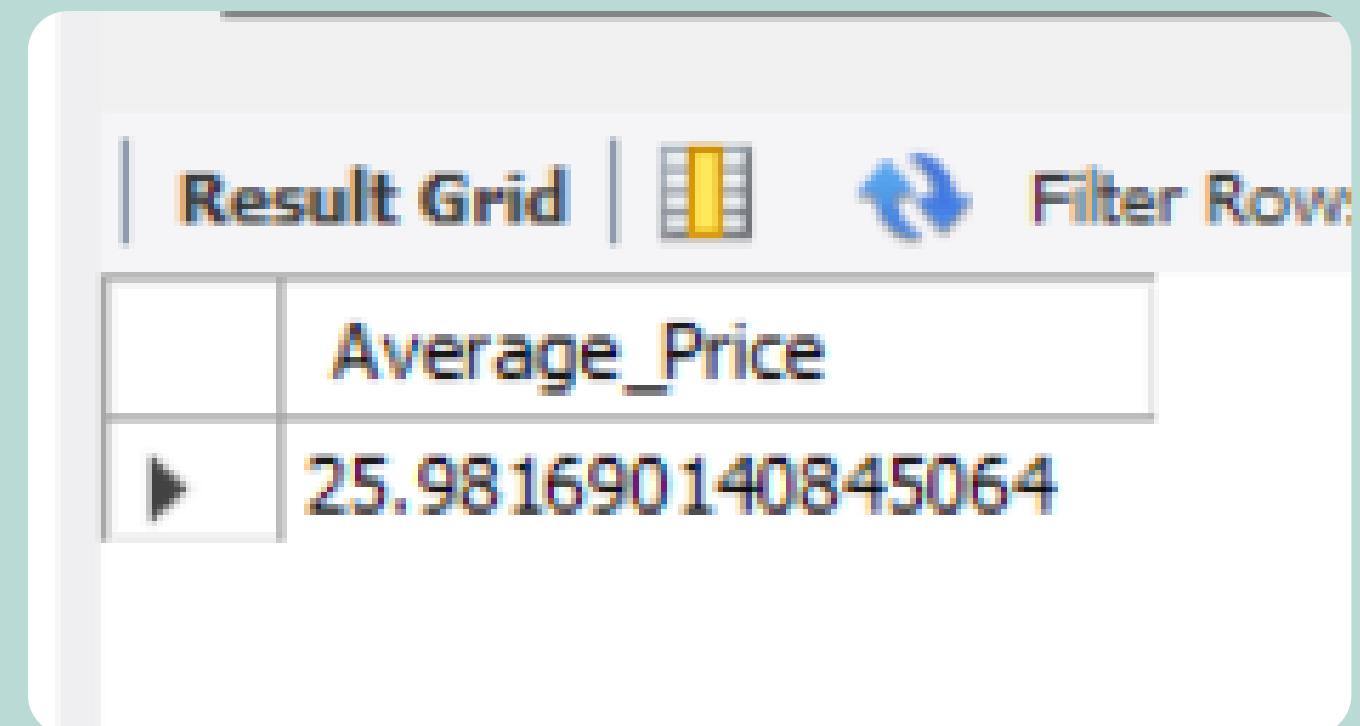


The screenshot shows a database query results grid with the following data:

	Genre	Total_Books_Sold
▶	Biography	285
	Non-Fiction	351
	Fantasy	446
	Romance	439
	Science Fiction	447
	Mystery	504
	Fiction	225

Find the average price of books in the 'Fantasy' genre.

```
SELECT AVG(Price) AS Average_Price  
FROM Books  
WHERE Genre='Fantasy';
```

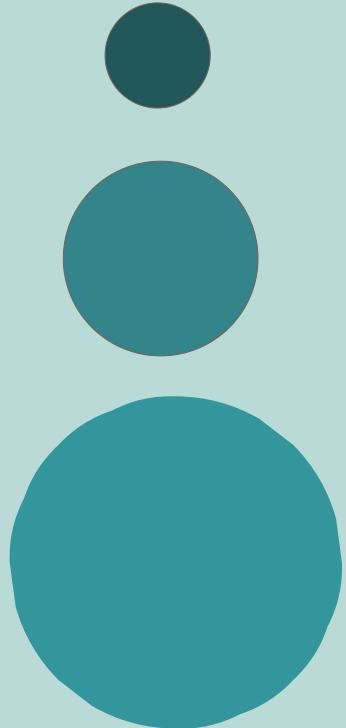


The screenshot shows a software interface for viewing database results. At the top, there are buttons for "Result Grid" (highlighted in blue), "Filter Row", and "Filter Column". The result grid displays a single row of data:

	Average_Price
▶	25.981690140845064

List customers who have placed atleast two orders.

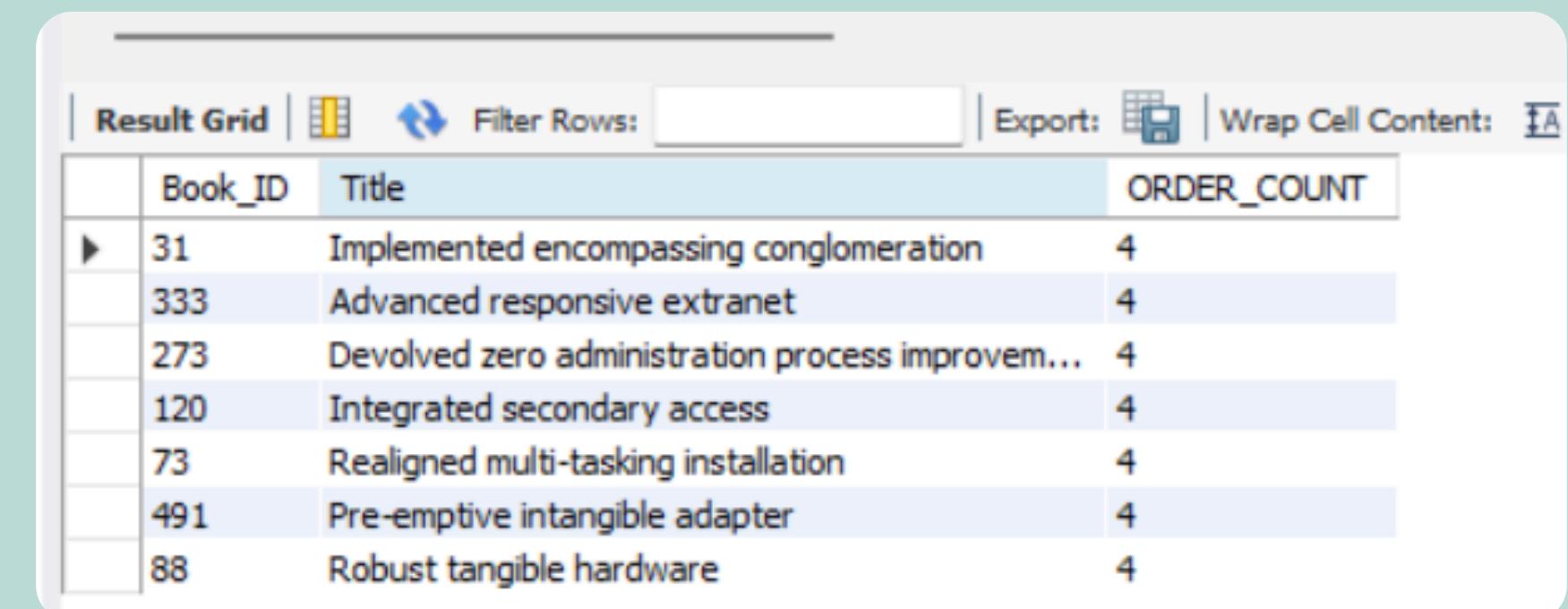
```
SELECT O.Customer_ID, C.Name, COUNT(O.Order_ID) AS ORDER_COUNT  
FROM Orders O  
JOIN Customers C ON O.Customer_ID = C.Customer_ID  
GROUP BY O.Customer_ID, C.Name  
HAVING COUNT(Order_ID) >=2;
```



	Customer_ID	Name	ORDER_COUNT
▶	2	Crystal Clements	2
	6	Stephen Vasquez	2
	8	Matthew Johnson	2
	13	Kristine Kim	2
	14	John Wood	2
	15	Vanessa Gaines	2
	16	Stacey Flores	3
	21	Edgar Frost	2
	22	Stacey Adams	3
	23	Howard Phillips	2

Find the most frequently ordered book.

```
SELECT O.Book_ID, B.Title, COUNT(O.Order_ID) AS ORDER_COUNT  
FROM Orders O  
JOIN Books B ON O.Book_ID = B.Book_ID  
GROUP BY O.Book_ID, B.Title  
ORDER BY ORDER_COUNT DESC LIMIT 7;
```

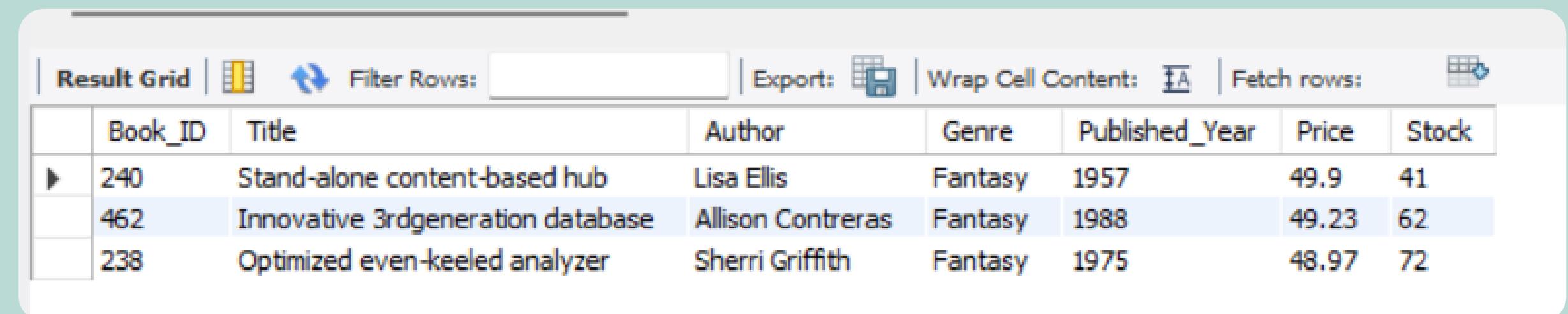


The screenshot shows a database query results grid with the following data:

	Book_ID	Title	ORDER_COUNT
▶	31	Implemented encompassing conglomeration	4
	333	Advanced responsive extranet	4
	273	Devolved zero administration process improvem...	4
	120	Integrated secondary access	4
	73	Realigned multi-tasking installation	4
	491	Pre-emptive intangible adapter	4
	88	Robust tangible hardware	4

Show the top 3 most expensive books of 'Fantasy' genre.

```
SELECT * FROM Books  
WHERE Genre = 'Fantasy'  
ORDER BY Price DESC LIMIT 3;
```



The screenshot shows a database query results grid with the following data:

	Book_ID	Title	Author	Genre	Published_Year	Price	Stock
▶	240	Stand-alone content-based hub	Lisa Ellis	Fantasy	1957	49.9	41
	462	Innovative 3rdgeneration database	Allison Contreras	Fantasy	1988	49.23	62
	238	Optimized even-keeled analyzer	Sherri Griffith	Fantasy	1975	48.97	72

Retrieve the total quantity of books sold by each author.

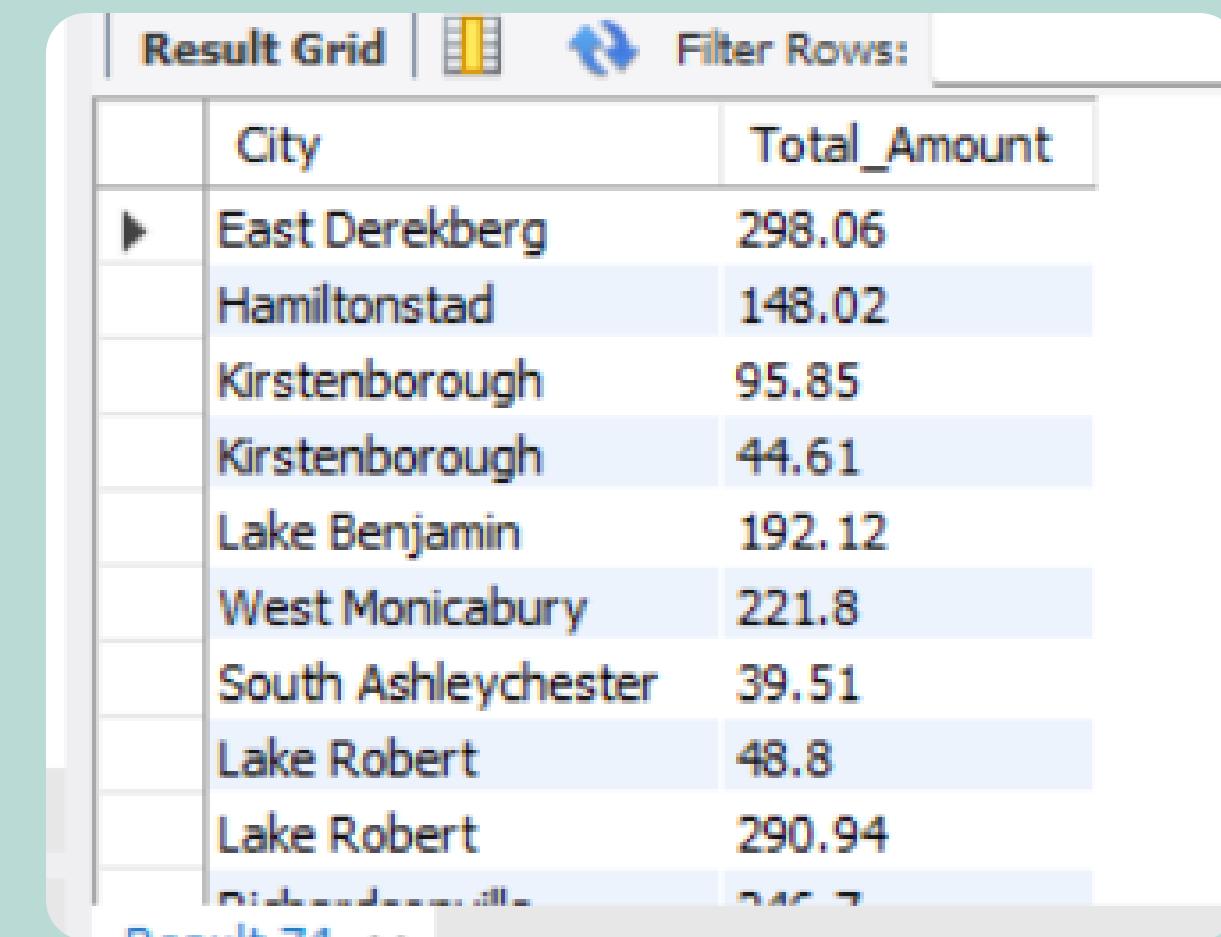
```
SELECT B.Author, SUM(O.Quantity) AS TOTAL_BOOKS_SOLD  
FROM Orders O  
JOIN Books B ON O.Book_ID = B.Book_ID  
GROUP BY B.Author;
```

Result Grid | Filter Rows:

	Author	TOTAL_BOOKS_SOLD
▶	Joseph Crane	3
	Derrick Howard	5
	Juan Miller	8
	Jacqueline Young	5
	Troy Cox	3
	Samantha Ruiz	1
	Denise Barnes	5
	Jadyn Miller	9
	Christopher Price	1
	Madeline Parker	0

List the cities where customers who spent over \$30 are located.

```
SELECT DISTINCT C.City, Total_Amount  
FROM Orders O  
JOIN Customers C ON O.Customer_ID = C.Customer_ID  
WHERE O.Total_Amount >30;
```

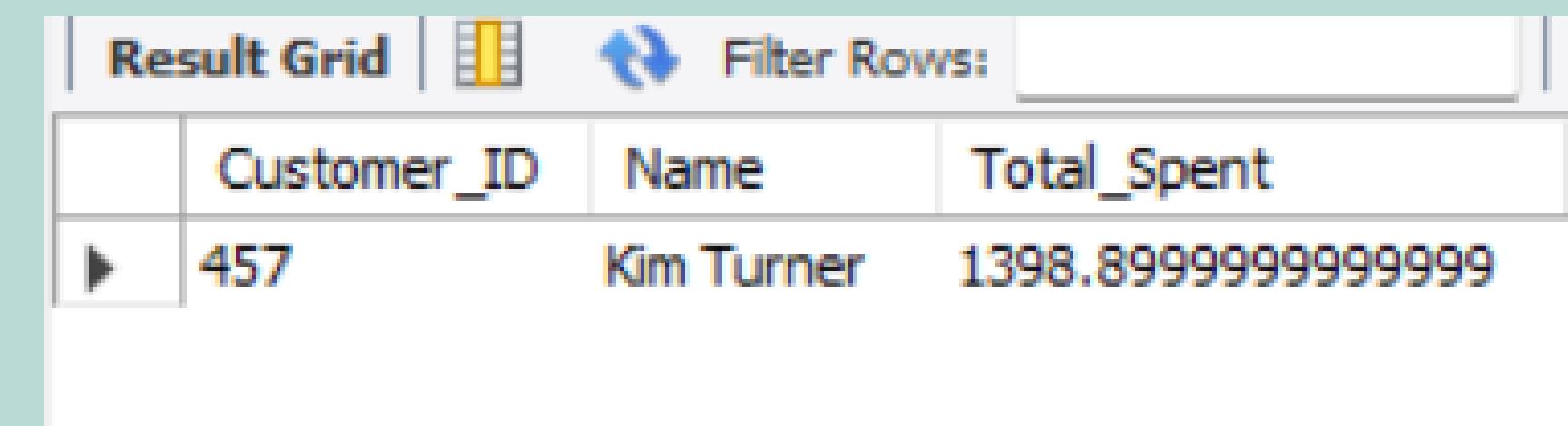


The screenshot shows a database query results grid titled "Result Grid". The grid has two columns: "City" and "Total_Amount". The data is as follows:

	City	Total_Amount
▶	East Derekberg	298.06
	Hamiltonstad	148.02
	Kirstenborough	95.85
	Kirstenborough	44.61
	Lake Benjamin	192.12
	West Monicabury	221.8
	South Ashleychester	39.51
	Lake Robert	48.8
	Lake Robert	290.94

Find the customer who spent the most on orders.

```
SELECT C.Customer_ID, C.Name, SUM(O.Total_Amount) AS Total_Spent  
FROM Orders O  
JOIN Customers C ON O.Customer_ID = C.Customer_ID  
GROUP BY C.Customer_ID, C.Name  
ORDER BY Total_Spent DESC LIMIT 1;
```



A screenshot of a database query results grid. The grid has a header row with four columns: Customer_ID, Name, and Total_Spent. The data row shows a single entry for customer ID 457, name Kim Turner, and total spent 1398.8999999999999.

	Customer_ID	Name	Total_Spent
▶	457	Kim Turner	1398.8999999999999

Calculate the stock remaining after fulfilling all orders.

```
SELECT B.Book_ID, B.Title, B.Stock, COALESCE(SUM(O.Quantity),0) AS Order_Qty,  
B.Stock - COALESCE(SUM(O.Quantity),0) AS Remaining_Qty  
FROM Books B  
LEFT JOIN Orders O ON B.Book_ID = O.Book_ID  
GROUP BY B.Book_ID,B.Title,B.Stock;
```

	Book_ID	Title	Stock	Order_Qty	Remaining_Qty
▶	1	Configurable modular throughput	100	3	97
	2	Persevering reciprocal knowledge user	19	0	19
	3	Streamlined coherent initiative	27	5	22
	4	Customizable 24hour product	8	0	8
	5	Adaptive 5thgeneration encoding	16	8	8
	6	Advanced encompassing implementation	2	0	2
	7	Open-architected exuding structure	95	5	90
	8	Persistent local encoding	84	3	81
	9	Optimized interactive challenge	70	0	70
	10	Configurable modular throughput	25	1	24

Thank You

