Certainly! Here’s a simple example using GROUP BY, HAVING, and ORDER BY without using joins. We'll use a single table to keep the example straightforward.

**Business Case Scenario**

**Scenario**: You are managing a small bookstore and want to analyze book sales data. Your goal is to summarize sales data by book title, filter results to show only those with sales above a certain threshold, and order the results by total sales.

**DDL (Data Definition Language)**

1. **Creating the Table**

sql

Copy code

CREATE TABLE book\_sales (

sale\_id SERIAL PRIMARY KEY,

book\_title VARCHAR(100) NOT NULL,

quantity\_sold INT NOT NULL,

sale\_date DATE NOT NULL

);

**Sample Data Insertion**

1. **Inserting Sample Data**

sql

Copy code

INSERT INTO book\_sales (book\_title, quantity\_sold, sale\_date) VALUES

('The Great Gatsby', 10, '2024-01-01'),

('1984', 5, '2024-01-05'),

('To Kill a Mockingbird', 8, '2024-01-10'),

('The Great Gatsby', 7, '2024-01-15'),

('1984', 10, '2024-01-20'),

('The Catcher in the Rye', 4, '2024-01-25'),

('To Kill a Mockingbird', 6, '2024-01-30');

**Queries**

1. **Using GROUP BY to Summarize Sales by Book Title**

sql

Copy code

SELECT book\_title,

SUM(quantity\_sold) AS total\_sold

FROM book\_sales

GROUP BY book\_title;

**Output**: This query summarizes the total quantity sold for each book title.

**Sample Output**

| **book\_title** | **total\_sold** |
| --- | --- |
| 1984 | 15 |
| The Catcher in the Rye | 4 |
| The Great Gatsby | 17 |
| To Kill a Mockingbird | 14 |

1. **Using HAVING to Filter Results**

To find books that have sold more than 10 copies:

sql

Copy code

SELECT book\_title,

SUM(quantity\_sold) AS total\_sold

FROM book\_sales

GROUP BY book\_title

HAVING SUM(quantity\_sold) > 10;

**Output**: This query filters the results to show only those books with total sales greater than 10.

**Sample Output**

| **book\_title** | **total\_sold** |
| --- | --- |
| 1984 | 15 |
| The Great Gatsby | 17 |
| To Kill a Mockingbird | 14 |

1. **Using ORDER BY to Sort the Results**

To order the results by total sales in descending order:

sql

Copy code

SELECT book\_title,

SUM(quantity\_sold) AS total\_sold

FROM book\_sales

GROUP BY book\_title

HAVING SUM(quantity\_sold) > 10

ORDER BY total\_sold DESC;

**Output**: This query shows books sold more than 10 copies, ordered by total sales from highest to lowest.

**Sample Output**

| **book\_title** | **total\_sold** |
| --- | --- |
| The Great Gatsby | 17 |
| To Kill a Mockingbird | 14 |
| 1984 | 15 |

**Summary**

* **GROUP BY**: Used to aggregate data based on the book\_title to calculate total sales.
* **HAVING**: Filters results to include only those books with total sales greater than 10.
* **ORDER BY**: Sorts the results based on the total quantity sold in descending order.

This example demonstrates how to use GROUP BY, HAVING, and ORDER BY in PostgreSQL without relying on joins, using a single table instead!