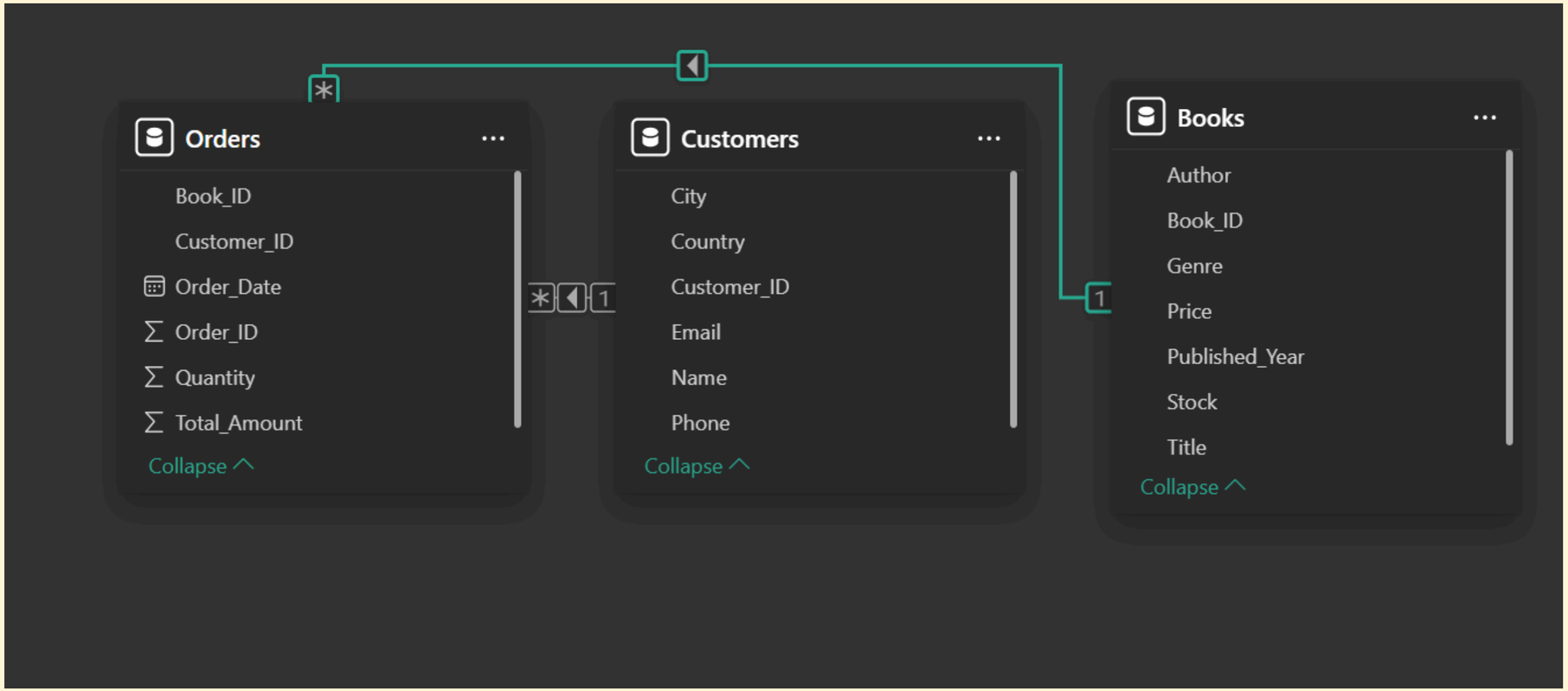
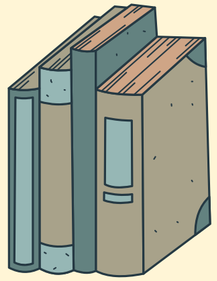
# ONLINE BOOKSTORE MANAGEMENT

## SQL PROJECT

Presented by: Aaditya Prajapat

# DATA MODEL


Orders
Book_ID
Customer_ID
Order_Date
Order_ID
Quantity
Total_Amount
Collapse
Customers
City
Country
Customer_ID
Email
Name
Phone
Collapse
Books
Author
Book_ID
Genre
Price
Published_Year
Stock
Title
Collapse



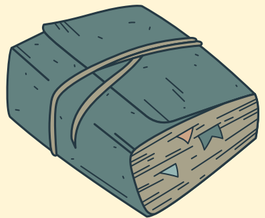
I'M AADITYA PRAJAPT, A PASSIONATE DATA ANALYTICS ENTHUSIAST WITH A DRIVE FOR TRANSFORMING RAW DATA INTO MEANINGFUL BUSINESS INSIGHTS. RECENTLY, I WORKED ON A HANDS-ON SQL PROJECT FOCUSED ON ONLINE BOOK STORE MANAGEMENT. 📚

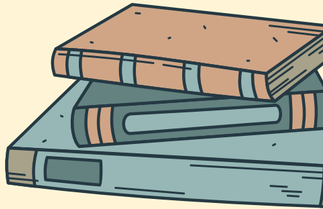
THIS PROJECT ENABLED ME TO:

- DESIGN AND MODEL A RELATIONAL DATABASE USING REAL-WORLD BUSINESS LOGIC.
- WRITE EFFICIENT SQL QUERIES TO EXTRACT DEEP INSIGHTS FROM CUSTOMER, ORDER, AND BOOK DATA.
- ANALYZE CUSTOMER BUYING BEHAVIOR, SALES TRENDS, INVENTORY PERFORMANCE, AND GENRE-WISE SALES DISTRIBUTION.
- DEVELOP A STRUCTURED, DATA-DRIVEN APPROACH TO IMPROVE OPERATIONAL EFFICIENCY AND DECISION-MAKING.

THROUGH THIS PROJECT, I EXPLORED THE FULL POTENTIAL OF SQL IN SOLVING PRACTICAL CHALLENGES, SUCH AS IDENTIFYING BEST-SELLING AUTHORS, TRACKING LOW-STOCK BOOKS, AND ANALYZING CUSTOMER DEMOGRAPHICS. EACH QUERY HELPED UNCOVER A NEW LAYER OF INSIGHT.

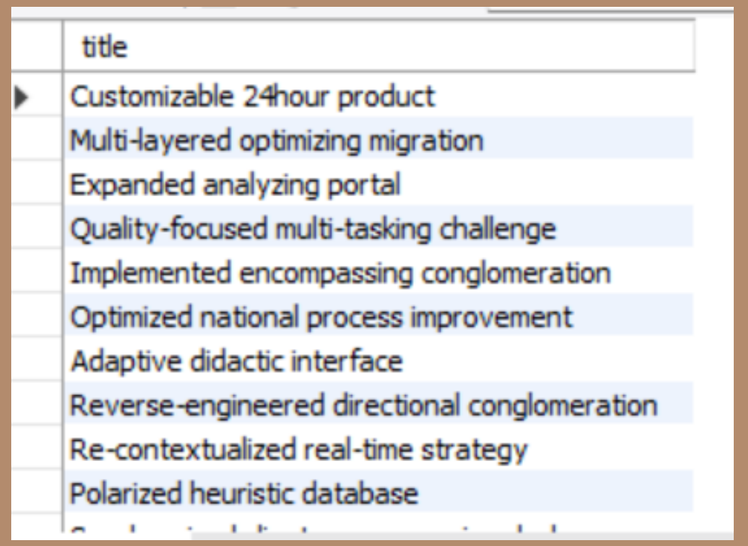
THIS EXPERIENCE SIGNIFICANTLY IMPROVED MY SKILLS IN SQL QUERYING, DATA MODELING, DATABASE RELATIONSHIPS, AND STORYTELLING WITH DATA—SKILLS THAT ARE CRITICAL FOR DATA-DRIVEN ROLES.





```sql
#1) Retrieve all books in the "Fiction" genre ?


SELECT
    title
FROM
    books
WHERE
    genre = 'Fiction';
```



| title |
| --- |
| Customizable 24hour product |
| Multi-layered optimizing migration |
| Expanded analyzing portal |
| Quality-focused multi-tasking challenge |
| Implemented encompassing conglomeration |
| Optimized national process improvement |
| Adaptive didactic interface |
| Reverse-engineered directional conglomeration |
| Re-contextualized real-time strategy |
| Polarized heuristic database |

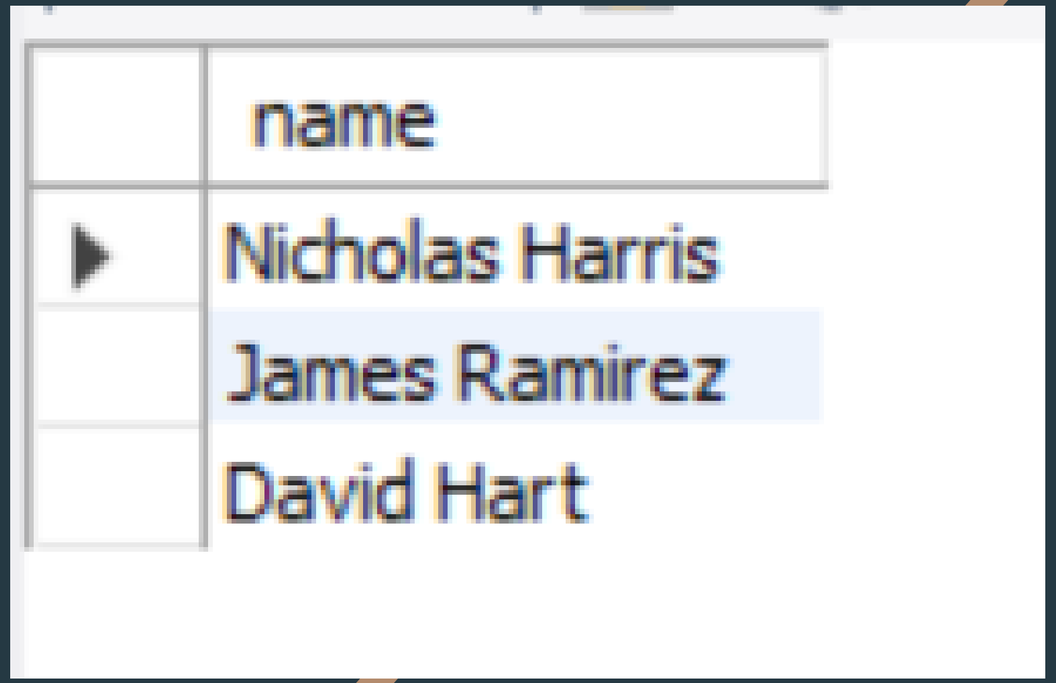
```sql
#2) Find books published after the year 1950

SELECT
    title, published_year
ROM
    books
HERE
    published_year > 1950;
```

| | title | published_year |
|---|---|---|
| ▶ | Persevering reciprocal knowledge user | 1971 |
| | Customizable 24hour product | 2020 |
| | Adaptive 5thgeneration encoding | 1956 |
| | Advanced encompassing implementation | 1985 |
| | Persistent local encoding | 2019 |
| | Optimized interactive challenge | 1987 |
| | Ergonomic national hub | 2015 |
| | Secured zero tolerance time-frame | 1998 |
| | Polarized optimal array | 1989 |
| | User-friendly motivating strategy | 1997 |

```sql
# 3) List all customers from the Canada ?

SELECT
    name
FROM
    customers
WHERE
    country = 'CANADA';
```



| name |
| --- |
| Nicholas Harris |
| James Ramirez |
| David Hart |

```sql
# 4) Show orders placed in November 2023 ?

SELECT
    Order_id, Customer_id, Book_id, order_date
FROM
    orders
WHERE
    Order_Date
```

| Order_id | Customer_id | Book_id | order_date |
|---|---|---|---|
| 1 | 84 | 169 | 2023-05-26 |
| 2 | 137 | 301 | 2023-01-23 |
| 3 | 216 | 261 | 2024-05-27 |
| 4 | 433 | 343 | 2023-11-25 |
| 5 | 14 | 431 | 2023-07-26 |
| 6 | 439 | 119 | 2024-10-11 |
| 7 | 195 | 467 | 2023-10-23 |
| 8 | 32 | 159 | 2024-05-07 |
| 9 | 109 | 407 | 2024-01-04 |
| 10 | 94 | 122 | 2024-07-09 |

```sql
#5) Retrieve the total stock of books available?


SELECT
    SUM(Stock) AS 'total stock'
FROM
    books;
```

| total stock |
| --- |
| 25056 |

```sql
# 6) Find the details of the most expensive book?

SELECT
    *
FROM
    books
ORDER BY price DESC
LIMIT 1;
```

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

```sql
#7) Show all customers who ordered more than 1 quantity of a boo

SELECT
    *
FROM
    orders
WHERE
    quantity > 1;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

| 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 | 94 | 122 | 2024-07-09 | 4 | 123 |

```sql
#8) Retrieve all orders where the total amount exceeds $20?

SELECT
    *
FROM
    orders
WHERE
    Total_amount > '$20';
```

| 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 | 94 | 122 | 2024-07-09 | 4 | 123 |

orders 62

#9) List all genres available in the Books table?

```sql
SELECT DISTINCT
    genre
FROM
    books;
```

| genre |
| --- |
| Biography |
| Fantasy |
| Non-Fiction |
| Fiction |
| Romance |
| Science Fiction |
| Mystery |

```sql
#10) Find the book with the lowest stock?

SELECT
    *
FROM
    books
ORDER BY stock
LIMIT 1;
```

| Book_ID | Title | Author | Genre | Published_Year | Price | Stock |
|---|---|---|---|---|---|---|
| 44 | Networked systemic implementation | Ryan Frank | Science Fiction | 1965 | 13.55 | 0 |

```sql
#11) Calculate the total revenue generated from all orders?

SELECT
    SUM(total_amount) AS 'Total Revenue Generated'
FROM
    orders;
```

| Total Revenue Generated |
| --- |
| 75628.66000000003 |

# ADVANCED QUERIES

```
#1) Retrieve the total number of books sold for each genre?

SELECT
    books.Genre, SUM(orders.Quantity) AS 'Total quantity sold'
FROM
    books
        INNER JOIN
    orders ON books.Book_ID = orders.Book_ID
GROUP BY genre;
```

| Genre | Total quantity sold |
|---|---|
| Biography | 285 |
| Non-Fiction | 351 |
| Fantasy | 446 |
| Romance | 439 |
| Science Fiction | 447 |
| Mystery | 504 |
| Fiction | 225 |

```
#3) List customers who have placed at least 2 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 | Hannah Drake | 2 |

```sql
#4) 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 1;
```

| Book_id | title | ORDER_COUNT |
|---|---|---|
| 31 | Implemented encompassing conglomeration | 4 |

```
#5) Show the top 3 most expensive books of 'Fantasy' Genre?

select * from books where Genre ='Fantasy' order by price desc limit 3 ;
```

| Book_ID | Title | Author | Genre | Published_Year | Price | Stock |
|---|---|---|---|---|---|---|
| 240 | Stand-alone content-based hub | Lisa Ellis | Fantasy | 1957 | 49.90 | 41 |
| 462 | Innovative 3rdgeneration database | Allison Contreras | Fantasy | 1988 | 49.23 | 62 |
| 238 | Optimized even-keeled analyzer | Sherri Griffith | Fantasy | 1975 | 48.97 | 72 |

```sql
#6) Retrieve the total quantity of books sold by each author ?

SELECT
    books.Author, SUM(orders.Quantity) AS 'total sold books'
FROM
    books
        JOIN
    orders ON books.Book_ID = orders.Book_ID
GROUP BY Author;
```

| Author | total sold books |
|---|---|
| Joseph Crane | 3 |
| Derrick Howard | 5 |
| Juan Miller | 8 |
| Jacqueline Young | 5 |
| Troy Cox | 3 |
| Samantha Ruiz | 1 |
| Denise Barnes | 5 |
| Jaclyn Miller | 9 |
| Christopher Price | 1 |
| Benjamin Peters | 9 |

Result 70

```
# 7) List the cities where customers who spent over $30 are located?

SELECT DISTINCT
    customers.city, orders.Total_Amount
FROM
    customers
        JOIN
    orders ON customers.Customer_ID = orders.Customer_ID
WHERE
    Total_Amount > '30';
```

| | 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 |
| | Richardsonville | 246.7 |

Result 71 ×

```sql
# 8) 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;
```

| customer_id | name | Total_Spent |
|---|---|---|
| 457 | Kim Turner | 1398.899999999999 |

# THANK YOU