

RETAIL SALES DATA ANALYSIS WITH SQL

Introduction

Welcome to the Retail Sales Data Analysis project! This document provides a comprehensive overview of how SQL was utilized to analyze retail sales data. The primary aim of this project is to demonstrate SQL skills through the exploration, cleaning, and analysis of data to answer crucial business questions.

The analysis encompasses setting up a retail sales database, performing exploratory data analysis (EDA), and deriving actionable insights to enhance business decision-making.

Requirements

- SQL Database (e.g., MySQL, PostgreSQL)
- SQL Client (e.g., MySQL Workbench, pgAdmin)

Project Overview

This project involves:

1. **Database Setup:** Establishing and populating a retail sales database with relevant tables and sample data.
2. **Data Exploration & Cleaning:** Conducting exploratory data analysis to understand and prepare the data for accurate analysis.
3. **Data Analysis & Findings:** Executing SQL queries to address specific business questions and derive valuable insights.

Business Questions

The following business questions guide the analysis of retail sales data:

1. Retrieve all columns for sales made on '2022-11-05'.
2. Retrieve all transactions where the category is 'Clothing' and the quantity sold is more than 4 in November 2022.
3. Calculate the total sales for each category.
4. Find the average age of customers who purchased items from the 'Beauty' category.
5. Find all transactions where the total sale amount is greater than 1000.
6. Find the total number of transactions made by each gender in each category.
7. Calculate the average sale for each month and identify the best-selling month in each year.
8. Find the top 5 customers based on the highest total sales.
9. Find the number of unique customers who purchased items from each category.
10. Create shifts (Morning, Afternoon, Evening) and count the number of orders per shift

Database Setup

- **Database Creation:** The project starts by creating a database named retailsales

```
create database retailsales;  
use retailsales;
```

Table Creation: A table named sales is created to store the sales data. The table structure includes columns for transaction ID, sale date, sale time, customer ID, gender, age, product category, quantity sold, price per unit, cost of goods sold (COGS), and total sale amount.

```
CREATE TABLE sales  
(  
    transactions_id INT PRIMARY KEY,  
    sale_date DATE,  
    sale_time TIME,  
    customer_id INT,  
    gender VARCHAR(10),  
    age INT,  
    category VARCHAR(35),  
    quantity INT,  
    price_per_unit FLOAT,  
    cogs FLOAT,  
    total_sale FLOAT  
);
```

Data Exploration & Cleaning

- **Record Count:** Determine the total number of records in the dataset.

```
SELECT COUNT(*) as total_records FROM sales;
```

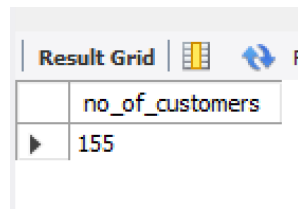
	total_records
▶	1987

- **Null Value Check:** Check for any null values in the dataset and delete records with missing data.

```
SELECT * FROM sales  
WHERE  
    sale_date IS NULL OR sale_time IS NULL OR customer_id IS NULL OR  
    gender IS NULL OR age IS NULL OR category IS NULL OR  
    quantity IS NULL OR price_per_unit IS NULL OR cogs IS NULL;
```

- **Customer Count:** Find out how many unique customers are in the dataset.

```
SELECT COUNT(DISTINCT customer_id) as no_of_customers FROM sales;
```

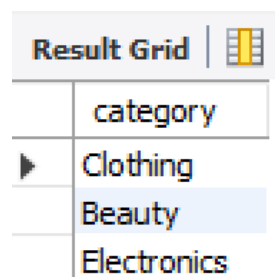


The screenshot shows a 'Result Grid' window with a single column header 'no_of_customers' and one data row containing the value '155'.

no_of_customers
155

- **Category Count:** Identify all unique product categories in the dataset.

```
SELECT DISTINCT category FROM sales;
```



The screenshot shows a 'Result Grid' window with a single column header 'category' and three data rows: 'Clothing', 'Beauty', and 'Electronics'.

category
Clothing
Beauty
Electronics

- **Additional Columns for Analysis**

Month: Extracted from the `sale_date` to allow for monthly analysis.

Year: Extracted from the `sale_date` to allow for yearly analysis.

```
alter table sales add column month varchar(10);
```

```
UPDATE sales
```

```
SET
```

```
month = MONTHNAME(sale_date);
```

```
alter table sales add column year int;
```

```
UPDATE sales
```

```
SET
```

```
year = YEAR(sale_date);
```

Business Questions and SQL Solutions

Question 1: Retrieve all columns for sales made on '2022-11-05'.

```
SELECT * FROM sales
WHERE sale_date = '2022-11-05';
```

transaction_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale	month	year
180	2022-11-05	10:47:00	117	Male	41	Clothing	3	300	129	900	November	2022
240	2022-11-05	11:49:00	95	Female	23	Beauty	1	300	123	300	November	2022
1256	2022-11-05	09:58:00	29	Male	23	Clothing	2	500	190	1000	November	2022
1587	2022-11-05	20:06:00	140	Female	40	Beauty	4	300	105	1200	November	2022
1819	2022-11-05	20:44:00	83	Female	35	Beauty	2	50	14	100	November	2022
943	2022-11-05	19:29:00	90	Female	57	Clothing	4	300	318	1200	November	2022
1896	2022-11-05	20:19:00	87	Female	30	Electronics	2	25	31	50	November	2022
1137	2022-11-05	22:34:00	104	Male	46	Beauty	2	500	145	1000	November	2022
856	2022-11-05	17:43:00	102	Male	54	Electronics	4	30	9	120	November	2022
214	2022-11-05	16:31:00	53	Male	20	Beauty	2	30	8	60	November	2022
1265	2022-11-05	14:35:00	86	Male	55	Clothing	3	300	111	900	November	2022

Question 2: Retrieve all transactions where the category is 'Clothing' and the quantity sold is more than 3 in November 2022.

```
SELECT * FROM sales
WHERE
    category = 'Clothing' AND quantity > 3
    AND month = 'November' AND year = 2022;
```

transaction_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale	month	year
1484	2022-11-23	09:29:00	22	Female	19	Clothing	4	300	147	1200	November	2022
64	2022-11-15	06:34:00	7	Male	49	Clothing	4	25	9	100	November	2022
284	2022-11-12	09:17:00	129	Male	43	Clothing	4	50	21	200	November	2022
1885	2022-11-09	07:32:00	148	Female	52	Clothing	4	30	11	120	November	2022
547	2022-11-14	07:36:00	3	Male	63	Clothing	4	500	250	2000	November	2022
159	2022-11-10	21:30:00	42	Male	26	Clothing	4	50	24	200	November	2022
699	2022-11-21	22:21:00	129	Female	37	Clothing	4	30	16	120	November	2022
1259	2022-11-03	17:31:00	105	Female	45	Clothing	4	50	21	200	November	2022
146	2022-11-10	22:01:00	74	Male	38	Clothing	4	50	49	200	November	2022
1476	2022-11-11	22:27:00	130	Female	27	Clothing	4	500	555	2000	November	2022
1296	2022-11-26	20:42:00	45	Female	22	Clothing	4	300	342	1200	November	2022
1696	2022-11-21	17:59:00	24	Female	50	Clothing	4	50	55	200	November	2022
1497	2022-11-19	21:44:00	109	Male	41	Clothing	4	30	32	120	November	2022
735	2022-11-26	21:38:00	153	Female	64	Clothing	4	500	515	2000	November	2022
943	2022-11-05	19:29:00	90	Female	57	Clothing	4	300	318	1200	November	2022
965	2022-11-27	21:45:00	84	Male	22	Clothing	4	50	13	200	November	2022
1615	2022-11-17	13:43:00	82	Female	61	Clothing	4	25	14	100	November	2022

sales 19 x

Question 3: Calculate the total sales for each category.

```
SELECT
    category, SUM(total_sale) AS total_sales
FROM sales
GROUP BY category;
```

category	total_sales
Clothing	309995
Beauty	286790
Electronics	311445

Question 4: Find the average age of customers who purchased items from the 'Beauty' category.

```
SELECT  
    ROUND(AVG(age)) AS average_age  
FROM sales  
WHERE category = 'Beauty';
```

Result Grid	
	average_age
▶	40

Question 5: Find all transactions where the total_sale amount is greater than 1000.

```
SELECT * FROM sales  
WHERE  
    total_sale > 1000;
```

Result Grid				Filter Rows:		Export:		Wrap Cell Content:		Fetch rows:			
	transaction_id	sale_date	sale_time	customer_id	gender	age	category	quantity	price_per_unit	cogs	total_sale	month	year
▶	522	2022-07-09	11:00:00	52	Male	46	Beauty	3	500	145	1500	July	2022
	559	2022-12-12	10:48:00	5	Female	40	Clothing	4	300	84	1200	December	2022
	1522	2022-11-14	08:35:00	48	Male	46	Beauty	3	500	235	1500	November	2022
	1559	2022-08-20	07:40:00	49	Female	40	Clothing	4	300	144	1200	August	2022
	421	2022-04-08	08:43:00	66	Female	37	Clothing	3	500	235	1500	April	2022
	1421	2022-01-17	07:07:00	59	Female	37	Clothing	3	500	185	1500	January	2022
	484	2022-03-13	07:52:00	135	Female	19	Clothing	4	300	75	1200	March	2022
	1484	2022-11-23	09:29:00	22	Female	19	Clothing	4	300	147	1200	November	2022
	15	2022-07-01	11:50:00	75	Female	42	Electronics	4	500	210	2000	July	2022
	743	2022-08-07	07:54:00	55	Female	34	Beauty	4	500	260	2000	August	2022
	1015	2022-03-09	11:53:00	94	Female	42	Electronics	4	500	200	2000	March	2022
	1743	2022-10-26	09:37:00	47	Female	34	Beauty	4	500	250	2000	October	2022
	742	2022-03-19	06:08:00	37	Female	38	Electronics	4	500	195	2000	March	2022
	1742	2022-11-22	08:25:00	18	Female	38	Electronics	4	500	220	2000	November	2022
	420	2022-01-02	10:53:00	28	Female	22	Clothing	4	500	200	2000	January	2022
	1420	2022-04-15	07:01:00	138	Female	22	Clothing	4	500	205	2000	April	2022
	592	2022-12-26	09:15:00	77	Female	46	Beauty	4	500	275	2000	December	2022

Question 6: Find the total number of transactions made by each gender in each category.

```
SELECT category, gender, COUNT(transaction_id) as count  
FROM sales  
GROUP BY category , gender  
ORDER BY category;
```

Result Grid		Filter Rows:
category	gender	count
▶ Beauty	Female	330
Beauty	Male	281
Clothing	Female	347
Clothing	Male	351
Electronics	Female	335
Electronics	Male	343

Question 7: Calculate the average sale for each month and identify the best-selling month in each year.

```
SELECT
    year, month, avg_sale
FROM
    (
        SELECT
            year, month,
            AVG(total_sale) as avg_sale,
            RANK() OVER(PARTITION BY year ORDER BY AVG(total_sale) DESC) as position
        FROM sales
        GROUP BY year, month
    ) as t1
WHERE position = 1;
```

	year	month	avg_sale
▶	2022	July	541.3415
	2023	February	535.5319

Question 8: Find the top 5 customers based on the highest total sales.

```
SELECT
    customer_id, SUM(total_sale) AS total_sales
FROM sales
GROUP BY customer_id
ORDER BY total_sales DESC
LIMIT 5;
```

	customer_id	total_sales
▶	3	38440
	1	30750
	5	30405
	2	25295
	4	23580

Question 9: Find the number of unique customers who purchased items from each category.

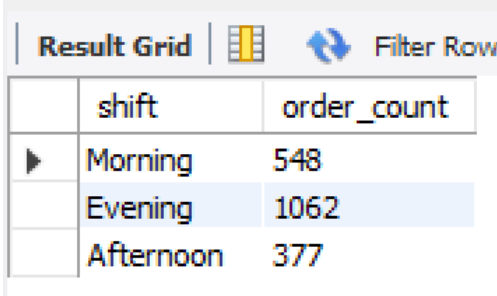
```
SELECT
    category, COUNT(DISTINCT customer_id) AS count
FROM sales
GROUP BY category;
```

	category	count
▶	Beauty	141
	Clothing	149
	Electronics	144

Question 10: Create shifts (Morning, Afternoon, Evening) and count the number of orders per shift.

SELECT CASE

```
    WHEN hour(sale_time) < 12 THEN 'Morning'
    WHEN hour(sale_time) BETWEEN 12 AND 17 THEN 'Afternoon'
    ELSE 'Evening'
END AS shift,
COUNT(transaction_id) AS order_count
FROM sales
GROUP BY shift;
```



	shift	order_count
▶	Morning	548
	Evening	1062
	Afternoon	377

Conclusion

In this document, we explored and analyzed retail sales data using SQL to answer various business questions. We set up a comprehensive database, performed exploratory data analysis, and addressed specific queries to derive meaningful insights.

Summary of Findings

- **Sales on Specific Dates:** We retrieved detailed sales data for specific dates, like '2022-11-05', and identified transactions in the 'Clothing' category with quantities over 4 in November 2022.
- **Total Sales by Category:** Calculated total sales for each category, revealing the most and least profitable categories.
- **Customer Demographics:** Determined the average age of customers purchasing from the 'Beauty' category.
- **High-Value Transactions:** Identified transactions where the total sale amount exceeded 1000.
- **Transaction Counts:** Analyzed the number of transactions by gender and category, and tracked monthly sales to find the best-selling months.
- **Top Customers:** Pinpointed the top 5 customers based on total sales.
- **Unique Customers by Category:** Found the number of unique customers for each category.
- **Order Shifts:** Created shifts (Morning, Afternoon, Evening) to count the number of orders during different times of the day.