

## SQL PROJECT

### Objectives

1. **Set up a retail sales database:** Create and populate a retail sales database with the sales data provided.
2. **Data Cleaning:** Identify and remove any records with missing or null values.
3. **Exploratory Data Analysis (EDA):** Perform basic exploratory data analysis to understand the dataset.
4. **Business Analysis:** Use SQL to answer specific business questions and derive insights from the sales data.

### Project Structure

#### 1. Database Setup

- **Database Creation:** The project starts by creating a database named `p1_retail_db`.
- **Table Creation:** A table named `retail_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 DATABASE p1_retail_db;
```

```
CREATE TABLE retail_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  
);
```

#### 2. Data Exploration & Cleaning

- **Record Count:** Determine the total number of records in the dataset.
- **Customer Count:** Find out how many unique customers are in the dataset.
- **Category Count:** Identify all unique product categories in the dataset.

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

```
SELECT COUNT(*) FROM retail_sales;
SELECT COUNT(DISTINCT customer_id) FROM retail_sales;
SELECT DISTINCT category FROM retail_sales;
```

```
SELECT * FROM retail_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;
```

```
DELETE FROM retail_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;
```

### 3. Data Analysis & Findings

The following SQL queries were developed to answer specific business questions:

1. **Write a SQL query to retrieve all columns for sales made on '2022-11-05':**

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

2. **Write a SQL query to retrieve all transactions where the category is 'Clothing' and the quantity sold is more than 4 in the month of Nov-2022:**

```
SELECT
    *
FROM retail_sales
WHERE
    category = 'Clothing'
    AND
    TO_CHAR(sale_date, 'YYYY-MM') = '2022-11'
AND
    quantity >= 4
```

3. **Write a SQL query to calculate the total sales (total\_sale) for each category.:**

```
SELECT
    category,
    SUM(total_sale) as net_sale,
```

```
    COUNT(*) as total_orders
FROM retail_sales
GROUP BY 1
```

4. **Write a SQL query to find the average age of customers who purchased items from the 'Beauty' category.:**

```
SELECT
    ROUND(AVG(age), 2) as avg_age
FROM retail_sales
WHERE category = 'Beauty'
```

5. **Write a SQL query to find all transactions where the total\_sale is greater than 1000.:**

```
SELECT * FROM retail_sales
WHERE total_sale > 1000
```

6. **Write a SQL query to find the total number of transactions (transaction\_id) made by each gender in each category.:**

```
SELECT
    category,
    gender,
    COUNT(*) as total_trans
FROM retail_sales
GROUP
    BY
        category,
        gender
ORDER BY 1
```

7. **Write a SQL query to calculate the average sale for each month. Find out best selling month in each year:**

```
SELECT
    year,
    month,
    avg_sale
FROM
    (
        SELECT
            EXTRACT(YEAR FROM sale_date) as year,
            EXTRACT(MONTH FROM sale_date) as month,
            AVG(total_sale) as avg_sale,
            RANK() OVER(PARTITION BY EXTRACT(YEAR FROM sale_date) ORDER BY AVG(total_sale)
            DESC) as rank
        FROM retail_sales
```

```
GROUP BY 1, 2
) as t1
WHERE rank = 1
```

8. Write a SQL query to find the top 5 customers based on the highest total sales :

```
SELECT
    customer_id,
    SUM(total_sale) as total_sales
FROM retail_sales
GROUP BY 1
ORDER BY 2 DESC
LIMIT 5
```

9. **Write a SQL query to find the number of unique customers who purchased items from each category.:**

```
SELECT
    category,
    COUNT(DISTINCT customer_id) as cnt_unique_cs
FROM retail_sales
GROUP BY category
```

10. **Write a SQL query to create each shift and number of orders (Example Morning <12, Afternoon Between 12 & 17, Evening >17):**

```
WITH hourly_sale
AS
(
    SELECT *,
        CASE
            WHEN EXTRACT(HOUR FROM sale_time) < 12 THEN 'Morning'
            WHEN EXTRACT(HOUR FROM sale_time) BETWEEN 12 AND 17 THEN 'Afternoon'
            ELSE 'Evening'
        END as shift
    FROM retail_sales
)
SELECT
    shift,
    COUNT(*) as total_orders
FROM hourly_sale
GROUP BY shift
```

## Conclusion

This project serves as a comprehensive introduction to SQL for data analysts, covering database setup, data cleaning, exploratory data analysis, and business-driven SQL

queries. The findings from this project can help drive business decisions by understanding sales patterns, customer behavior, and product performance.