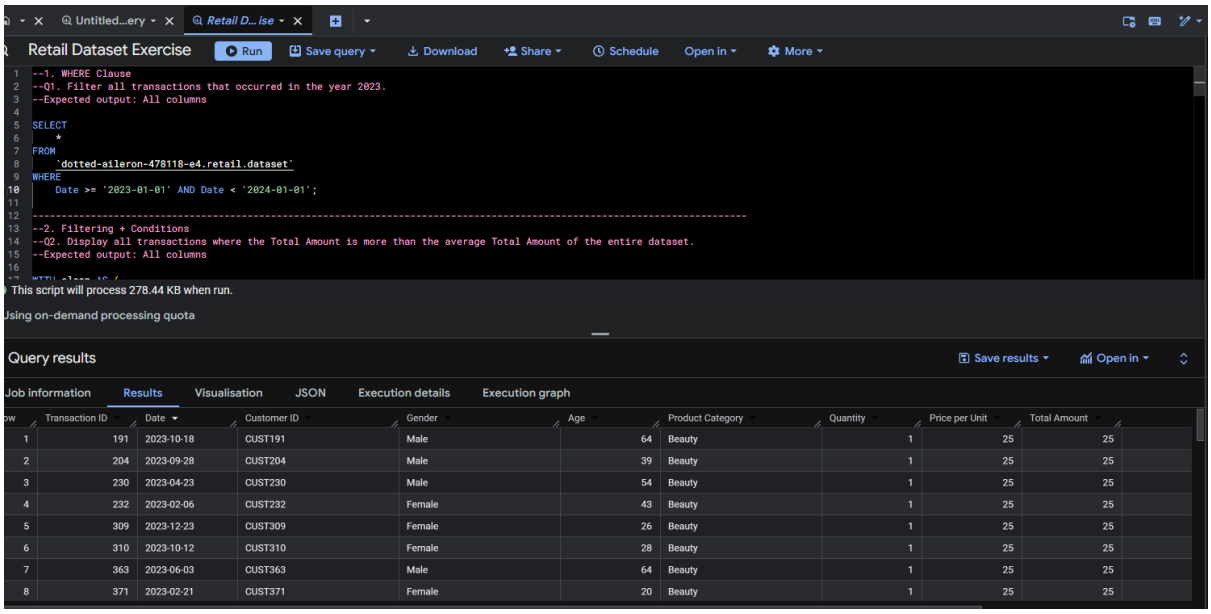
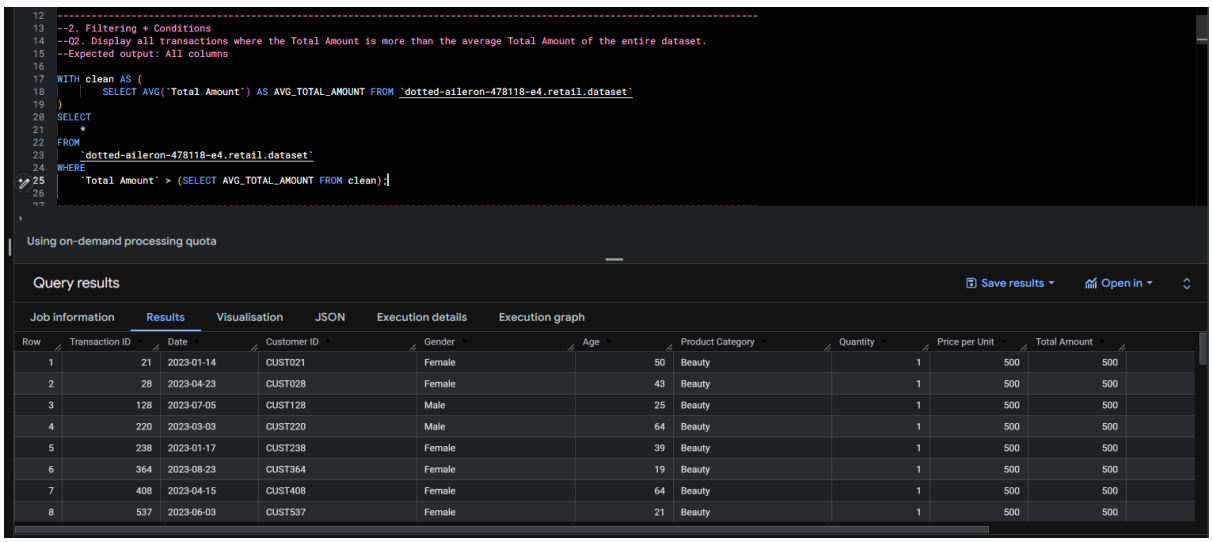
# BIG QUERY PRACTICAL EXERCISE

## Question 1




| Row | Transaction ID | Date | Customer ID | Gender | Age | Product Category | Quantity | Price per Unit | Total Amount |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 191 | 2023-10-18 | CUST191 | Male | 64 | Beauty | 1 | 25 | 25 |
| 2 | 204 | 2023-09-28 | CUST204 | Male | 39 | Beauty | 1 | 25 | 25 |
| 3 | 230 | 2023-04-23 | CUST230 | Male | 54 | Beauty | 1 | 25 | 25 |
| 4 | 232 | 2023-02-06 | CUST232 | Female | 43 | Beauty | 1 | 25 | 25 |
| 5 | 309 | 2023-12-23 | CUST309 | Female | 26 | Beauty | 1 | 25 | 25 |
| 6 | 310 | 2023-10-12 | CUST310 | Female | 28 | Beauty | 1 | 25 | 25 |
| 7 | 363 | 2023-06-03 | CUST363 | Male | 64 | Beauty | 1 | 25 | 25 |
| 8 | 371 | 2023-02-21 | CUST371 | Female | 20 | Beauty | 1 | 25 | 25 |

## Question 2




| Row | Transaction ID | Date | Customer ID | Gender | Age | Product Category | Quantity | Price per Unit | Total Amount |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 21 | 2023-01-14 | CUST021 | Female | 50 | Beauty | 1 | 500 | 500 |
| 2 | 28 | 2023-04-23 | CUST028 | Female | 43 | Beauty | 1 | 500 | 500 |
| 3 | 128 | 2023-07-05 | CUST128 | Male | 25 | Beauty | 1 | 500 | 500 |
| 4 | 220 | 2023-03-03 | CUST220 | Male | 64 | Beauty | 1 | 500 | 500 |
| 5 | 238 | 2023-01-17 | CUST238 | Female | 39 | Beauty | 1 | 500 | 500 |
| 6 | 364 | 2023-08-23 | CUST364 | Female | 19 | Beauty | 1 | 500 | 500 |
| 7 | 408 | 2023-04-15 | CUST408 | Female | 64 | Beauty | 1 | 500 | 500 |
| 8 | 537 | 2023-06-03 | CUST537 | Female | 21 | Beauty | 1 | 500 | 500 |

# Question 3

```sql
SELECT SUM(`Total Amount`) AS Total_Revenue
FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;

-----------------------------------------------------------------------------------------------------------------------
--4. DISTINCT
--Q4. Display all distinct Product Categories in the dataset.
--Expected output: Product_Category

SELECT DISTINCT `Product Category`
FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;
```

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Total_Revenue |
|---|---|
| 1 | 456000 |

# Question 4

```sql
--4. DISTINCT
--Q4. Display all distinct Product Categories in the dataset.
--Expected output: Product_Category

SELECT DISTINCT `Product Category`
FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;
```

Query completed

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Product Category |
|---|---|
| 1 | Beauty |
| 2 | Clothing |
| 3 | Electronics |

# Question 5

```sql
SELECT `Product Category`, SUM(Quantity) AS Total_Quantity
FROM `dotted-aileron-478118-e4.retail.dataset`
GROUP BY `Product Category`;

-----------------------------------------------------------------------------------------------------------------------
--6. CASE Statement
--Q6. Create a column called Age_Group that classifies customers as 'Youth' (<30), 'Adult' (30–59), and 'Senior' (60+).
--Expected output: Customer ID, Age, Age Group
```

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Product Category | Total_Quantity |
|---|---|---|
| 1 | Beauty | 771 |
| 2 | Clothing | 894 |
| 3 | Electronics | 849 |

# Question 6

```sql
--6. CASE Statement
--Q6. Create a column called Age_Group that classifies customers as 'Youth' (<30), 'Adult' (30-59), and 'Senior' (60+).
--Expected output: Customer_ID, Age, Age_Group

SELECT
  `Customer ID`,
   Age,
  CASE WHEN Age < 30 THEN 'Youth'
       WHEN Age BETWEEN 30 AND 59 THEN 'Adult'
       ELSE 'Senior'
  END AS Age_Group
FROM `dotted-aileron-478118-e4.retail.dataset`;

--7. Conditional Aggregation
```

Query completed

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Customer ID | Age | Age_Group |
|---|---|---|---|
| 1 | CUST191 | 64 | Senior |
| 2 | CUST204 | 39 | Adult |
| 3 | CUST230 | 54 | Adult |
| 4 | CUST232 | 43 | Adult |
| 5 | CUST309 | 26 | Youth |
| 6 | CUST310 | 28 | Youth |
| 7 | CUST363 | 64 | Senior |
| 8 | CUST371 | 20 | Youth |

# Question 7

```sql
--7. Conditional Aggregation
--Q7. For each Gender, count how many high-value transactions occurred (where Total Amount > 500).
--Expected output: Gender, High_Value_Transactions

SELECT Gender,
       COUNT(`Transaction ID`) AS High_Value_Transactions
FROM `dotted-aileron-478118-e4.retail.dataset`
WHERE `Total Amount` > 500
GROUP BY Gender;

--8. HAVING Clause
--Q8. For each Product Category, show only those categories where the total revenue exceeds 5,000.
--Expected output: Product_Category, Total_Revenue
```

Query completed

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Gender | High_Value_Tran... |
|---|---|---|
| 1 | Female | 155 |
| 2 | Male | 144 |

# Question 8

```sql
--8. HAVING Clause
--Q8. For each Product Category, show only those categories where the total revenue exceeds 5,000.
--Expected output: Product_Category, Total_Revenue

SELECT `Product Category`,
       SUM(`Total Amount`) As Total_Revenue
FROM `dotted-aileron-478118-e4.retail.dataset`
GROUP BY `Product Category`
HAVING SUM(`Total Amount`) > 5000;

--Q9. Display a new column called Unit_Cost_Category that labels a transaction as: – 'Cheap' if Price per Unit < 50 – 'Moderate' if Price per Unit between 50 and 200 – 'Expensive' if Price per Unit > 200
--Expected output: Transaction_ID, Price_per_Unit, Unit_Cost_Category
```

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Product Category | Total_Revenue |
|---|---|---|
| 1 | Beauty | 143515 |
| 2 | Clothing | 155580 |
| 3 | Electronics | 156905 |

# Question 9

```sql
--Q9. Display a new column called Unit_Cost_Category that labels a transaction as: – 'Cheap' if Price per Unit < 50 – 'Moderate' if Price per Unit between 50 and 200 – 'Expensive' if Price per Unit
> 200
--Expected output: Transaction_ID, Price_per_Unit, Unit_Cost_Category

SELECT `TRANSACTION ID`,
       `price per unit`,
   CASE WHEN `price per unit` < 50 THEN 'Cheap'
        WHEN `price per unit` BETWEEN 50 AND 200 THEN 'Moderate'
        ELSE 'Expensive'
   END AS Unit_Price_Category
FROM `dotted-aileron-478118-e4.retail.dataset`;

--10. Combining WHERE + CASE
```

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | TRANSACTION ID | price per unit | Unit_Price_Category |
|---|---|---|---|
| 1 | 191 | 25 | Cheap |
| 2 | 204 | 25 | Cheap |
| 3 | 230 | 25 | Cheap |
| 4 | 232 | 25 | Cheap |
| 5 | 309 | 25 | Cheap |
| 6 | 310 | 25 | Cheap |
| 7 | 363 | 25 | Cheap |
| 8 | 371 | 25 | Cheap |

# Question 10

```sql
--10. Combining WHERE + CASE


SELECT `Customer ID`,
       Age,
       `Total Amount`,
     CASE WHEN `Total Amount` > 1000 THEN 'High'
          ELSE 'Low'
     END AS Spending_Level
FROM `dotted-aileron-478118-e4.retail.dataset`
WHERE Age >= 40;

```

This script will process 278.44 KB when run.

Using on-demand processing quota

Query results

Save results | Open in

Job information | Results | Visualisation | JSON | Execution details | Execution graph

| Row | Customer ID | Age | Total Amount | Spending_Level |
|---|---|---|---|---|
| 1 | CUST191 | 64 | 25 | Low |
| 2 | CUST230 | 54 | 25 | Low |
| 3 | CUST232 | 43 | 25 | Low |
| 4 | CUST363 | 64 | 25 | Low |
| 5 | CUST454 | 46 | 25 | Low |
| 6 | CUST512 | 57 | 25 | Low |
| 7 | CUST791 | 51 | 25 | Low |
| 8 | CUST825 | 46 | 25 | Low |