

# BIG QUERY PRACTICAL EXERCISE

## Question 1

The screenshot shows the BigQuery interface with a query titled "Retail Dataset Exercise". The code is as follows:

```
1 --1. WHERE Clause
2 --Q1. Filter all transactions that occurred in the year 2023.
3 --Expected output: All columns
4
5 SELECT *
6 FROM
7 `dotted-aileron-478118-e4.retail.dataset`
8 WHERE
9     Date >= '2023-01-01' AND Date < '2024-01-01';
10
11 -----
12 --2. Filtering + Conditions
13 --Q2. Display all transactions where the Total Amount is more than the average Total Amount of the entire dataset.
14 --Expected output: All columns
15
16 -----
17 WITH clean AS (
18     SELECT AVG("Total Amount") AS AVG_TOTAL_AMOUNT FROM `dotted-aileron-478118-e4.retail.dataset`
19 )
20 SELECT *
21 FROM
22 `dotted-aileron-478118-e4.retail.dataset`
23 WHERE
24     `Total Amount` > (SELECT AVG.TOTAL_AMOUNT FROM clean);
25
26
```

This script will process 278.44 KB when run.

Using on-demand processing quota

Query results

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

The screenshot shows the BigQuery interface with a query for Question 2. The code is as follows:

```
12 -----
13 --2. Filtering + Conditions
14 --Q2. Display all transactions where the Total Amount is more than the average Total Amount of the entire dataset.
15 --Expected output: All columns
16
17 WITH clean AS (
18     SELECT AVG("Total Amount") AS AVG_TOTAL_AMOUNT FROM `dotted-aileron-478118-e4.retail.dataset`
19 )
20 SELECT *
21 FROM
22 `dotted-aileron-478118-e4.retail.dataset`
23 WHERE
24     `Total Amount` > (SELECT AVG.TOTAL_AMOUNT FROM clean);
25
26
```

This script will process 278.44 KB when run.

Using on-demand processing quota

Query results

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

```
31
32   SELECT SUM('Total_Amount') AS Total_Revenue
33   FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;
34
35   -----
36   --4. DISTINCT
37   --Q4. Display all distinct Product Categories in the dataset.
38   --Expected output: Product_Category
39
40   SELECT DISTINCT 'Product_Category'
41   FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;
42
```

Query results

Job information	Results	Visualisation	JSON	Execution details	Execution graph
Row	Total_Revenue				
1	456000				

## Question 4

```
36   --4. DISTINCT
37   --Q4. Display all distinct Product Categories in the dataset.
38   --Expected output: Product_Category
39
40   SELECT DISTINCT 'Product_Category'
41   FROM `dotted-aileron-478118-e4`.`retail`.`dataset`;
42
```

Query completed  
Using on-demand processing quota

Query results

Job information	Results	Visualisation	JSON	Execution details	Execution graph
Row	Product Category				
1	Beauty				
2	Clothing				
3	Electronics				

## Question 5

```
48   SELECT 'Product Category', SUM(Quantity) AS Total_Quantity
49   FROM `dotted-aileron-478118-e4`.`retail`.`dataset`
50   GROUP BY 'Product Category';
51
52   -----
53   --6. CASE Statement
54   --Q6. Create a column called Age_Group that classifies customers as 'Youth' (<30), 'Adult' (30-59), and 'Senior' (60+).
55   --Expected output: Customer ID and Age Group
56
```

Using on-demand processing quota

Query results

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

```
52 -----  
53 --6. CASE Statement  
54 --Q6. Create a column called Age_Group that classifies customers as 'Youth' (<30), 'Adult' (30-59), and 'Senior' (60+).  
55 --Expected output: Customer_ID, Age, Age_Group  
56  
57 SELECT  
58     Customer_ID,  
59     Age,  
60     CASE WHEN Age < 30 THEN 'Youth'  
61       WHEN Age BETWEEN 30 AND 59 THEN "Adult"  
62       ELSE 'Senior'  
63     END AS Age_Group  
64 FROM 'dotted-aleron-478118-e4.retail.dataset';  
65  
66 -----  
67 --7. Conditional Aggregation  
68  
69 Query completed  
Using on-demand processing quota  
  
Query results  


| 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     |                   |                 |
| 9               | CUST397     | 50            | Adult     |                   |                 |


```

## Question 7

```
66 -----  
67 --7. Conditional Aggregation  
68 --Q7. For each Gender, count how many high-value transactions occurred (where Total_Amount > 500).  
69 --Expected output: Gender, High_Value_Transactions  
70  
71 SELECT Gender,  
72     COUNT(Transaction_ID) AS High_Value_Transactions  
73 FROM 'dotted-aleron-478118-e4.retail.dataset'  
74 WHERE Total_Amount > 500  
75 GROUP BY Gender;  
76  
77 -----  
78 --8. HAVING Clause  
79 --Q8. For each Product Category, show only those categories where the total revenue exceeds 5,000.  
80 --Expected output: Product_Cat..._Total_Revenue  
81  
82 Query completed  
Using on-demand processing quota  
  
Query results  


| Job information | Results | Visualisation      | JSON | Execution details | Execution graph |
|-----------------|---------|--------------------|------|-------------------|-----------------|
| Row             | Gender  | High_Value_Tran... |      |                   |                 |
| 1               | Female  | 155                |      |                   |                 |
| 2               | Male    | 144                |      |                   |                 |


```

## Question 8

```
77 -----  
78 --8. HAVING Clause  
79 --Q8. For each Product Category, show only those categories where the total revenue exceeds 5,000.  
80 --Expected output: Product_Cat..._Total_Revenue  
81  
82 SELECT 'Product Category',  
83     SUM(Total_Amount) AS Total_Revenue  
84 FROM 'dotted-aleron-478118-e4.retail.dataset'  
85 GROUP BY 'Product Category'  
86 HAVING SUM(Total_Amount) > 5000;  
87  
88 -----  
89 --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  
89 > 200  
90 --Expected output: Transaction_ID, Price_per_Unit, Unit_Cost_Category  
91  
92  
93 Query completed  
Using on-demand processing quota  
  
Query results  


| 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

```
88 --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
89 > 200
90 --Expected output: Transaction_ID, Price_per_Unit, Unit_Cost_Category
91
92     SELECT 'TRANSACTION ID',
93           'price per unit',
94           CASE WHEN 'price per unit' < 50 THEN 'Cheap'
95                 WHEN 'price per unit' BETWEEN 50 AND 200 THEN 'Moderate'
96                 ELSE 'Expensive'
97           END AS Unit_Price_Category
98   FROM `dotted-aieron-478118-e4.retail.dataset`;
99
100
101 --10. Combining WHERE + CASE
102
103
104 Using on-demand processing quota
```

Query results

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

```
100
101 --10. Combining WHERE + CASE
102
103
104     SELECT 'Customer ID',
105           'Age',
106           'Total Amount',
107           CASE WHEN 'Total Amount' > 1000 THEN 'High'
108                 ELSE "Low"
109           END AS Spending_Level
110   FROM `dotted-aieron-478118-e4.retail.dataset`
111 WHERE Age >= 40;
112
113
114
115
116 This script will process 278.44 KB when run.
```

Using on-demand processing quota

Query results

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	