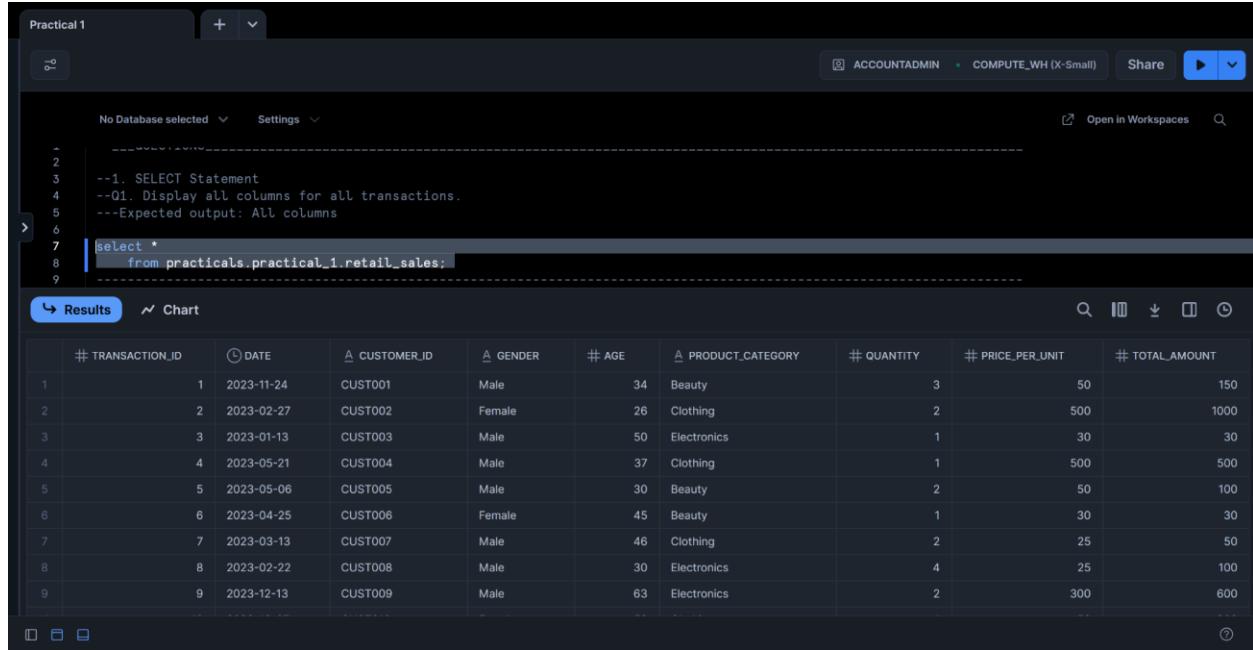


NTSHEMBO MALUEKE

Practical 1: SQL Fundamentals (Snowflake-Basic SQL Syntax)

QUESTIONS

Q1



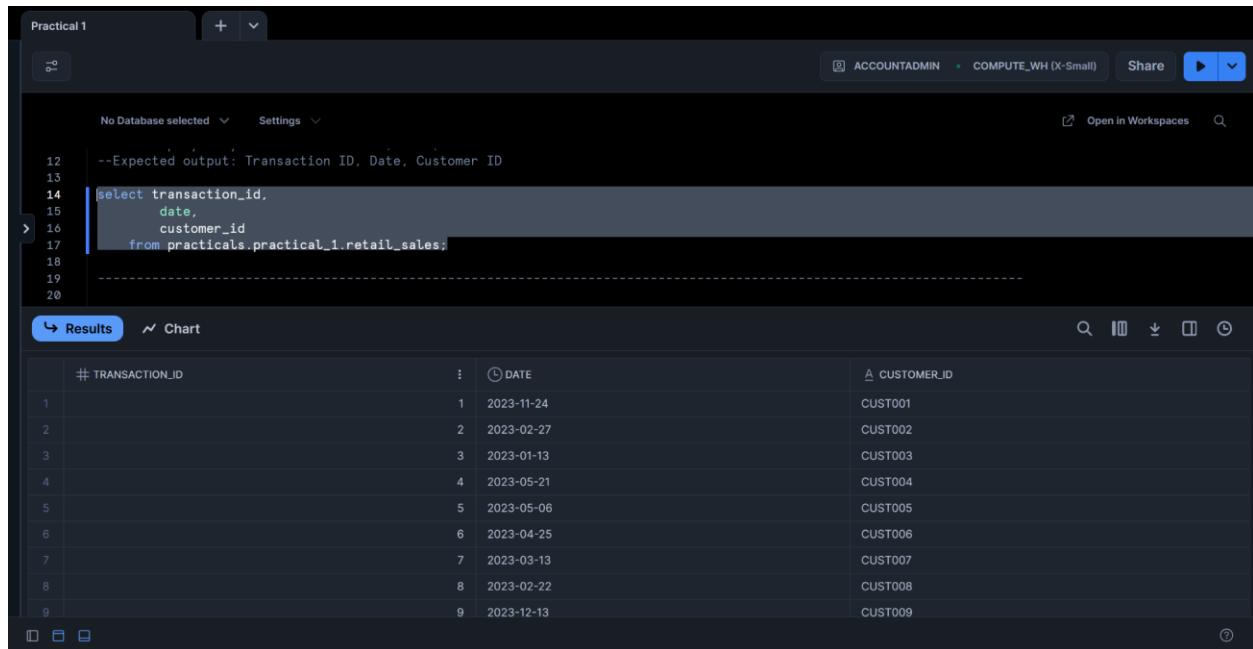
The screenshot shows the Snowflake interface with a dark theme. The top navigation bar includes 'Practical 1', a '+' button, account information ('ACCOUNTADMIN COMPUTE_WH (X-Small)'), 'Share' and 'Run' buttons, and links to 'Open in Workspaces' and a search bar. The main area displays a SQL query and its results.

```
--1. SELECT Statement
--Q1. Display all columns for all transactions.
---Expected output: All columns
select *
from practicals.practical_1.retail_sales;
```

The results table has the following columns and data:

	# TRANSACTION_ID	DATE	A CUSTOMER_ID	A GENDER	# AGE	A PRODUCT_CATEGORY	# QUANTITY	# PRICE_PER_UNIT	# TOTAL_AMOUNT
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
2	2	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
3	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
4	4	2023-05-21	CUST004	Male	37	Clothing	1	500	500
5	5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
6	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
7	7	2023-03-13	CUST007	Male	46	Clothing	2	25	50
8	8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
9	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600

Q2



The screenshot shows the Snowflake interface with a dark theme. The top navigation bar includes 'Practical 1', a '+' button, account information ('ACCOUNTADMIN COMPUTE_WH (X-Small)'), 'Share' and 'Run' buttons, and links to 'Open in Workspaces' and a search bar. The main area displays a SQL query and its results.

```
--Expected output: Transaction ID, Date, Customer ID
select transaction_id,
date,
customer_id
from practicals.practical_1.retail_sales;
```

The results table has the following columns and data:

	# TRANSACTION_ID	DATE	A CUSTOMER_ID
1	1	2023-11-24	CUST001
2	2	2023-02-27	CUST002
3	3	2023-01-13	CUST003
4	4	2023-05-21	CUST004
5	5	2023-05-06	CUST005
6	6	2023-04-25	CUST006
7	7	2023-03-13	CUST007
8	8	2023-02-22	CUST008
9	9	2023-12-13	CUST009

Q3

The screenshot shows a database query interface with the following details:

Practical 1 tab is selected.

No Database selected.

Settings dropdown is open.

Share button is present.

Open in Workspaces button is present.

Query code (lines 17-28):

```
17   from practicals.practical_1.retail_sales;
18
19
20 --Q3. Display all the distinct product categories in the dataset.
21 --Expected output: Product Category
22
23 select distinct(PRODUCT_CATEGORY) as Product_Category
24   from practicals.practical_1.retail_sales;
```

Results tab is selected.

Header: ▲ PRODUCT_CATEGORY

Data:

PRODUCT_CATEGORY
Clothing
Beauty
Electronics

Q4

The screenshot shows a database query interface with the following details:

Practical 1 tab is selected.

No Database selected.

Settings dropdown is open.

Share button is present.

Open in Workspaces button is present.

Query code (lines 26-58):

```
26
27 --Q4. Display all the distinct gender values in the dataset.
28 --Expected output: Gender
29
30
31 select distinct(Gender) as GENDER
32   from practicals.practical_1.retail_sales;
```

Results tab is selected.

Header: ▲ GENDER

Data:

GENDER
Male
Female

Q5

Practical 1

```
--Q5. Display all transactions where the Age is greater than 40.  
--Expected output: All columns  
37  
38 select *  
39   from practicals.practical_1.retail_sales  
40   where age > 40;  
41  
42
```

Results

# TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2023-01-13	CUST003	Male	50	Electronics	1	30	30
2	2023-04-25	CUST006	Female	45	Beauty	1	30	30
3	2023-03-13	CUST007	Male	46	Clothing	2	25	50
4	2023-12-13	CUST009	Male	63	Electronics	2	300	600
5	2023-10-07	CUST010	Female	52	Clothing	4	50	200
6	2023-01-17	CUST014	Male	64	Clothing	4	30	120
7	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
8	2023-04-30	CUST018	Female	47	Electronics	2	25	50

Q6

Practical 1

```
--Q6. Display all transactions where the Price per Unit is between 100 and 500.  
--Expected output: All columns  
45  
46 select *  
47   from practicals.practical_1.retail_sales  
48   where price_per_unit between 100 and 500;  
49  
50
```

Results

# TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TRANSACTION_ID
1	2023-02-27	CUST002	Female	26	Clothing	2	500	01bf...
2	2023-05-21	CUST004	Male	37	Clothing	1	500	01bf...
3	2023-12-13	CUST009	Male	63	Electronics	2	300	01bf...
4	2023-08-05	CUST013	Male	22	Electronics	3	500	01bf...
5	2023-01-16	CUST015	Female	42	Electronics	4	500	01bf...
6	2023-02-17	CUST016	Male	19	Clothing	3	500	01bf...
7	2023-11-05	CUST020	Male	22	Clothing	3	300	01bf...
8	2023-01-14	CUST021	Female	50	Beauty	1	500	01bf...

Query Details

- Query duration: 39ms
- Rows: 396
- Query ID: 01bf...

TRANSACTION_ID

Q7

Practical 1

```
50  -----
51  --Q7. Display all transactions where the Product Category is either 'Beauty' or
52  --'Electronics'.
53
54  select *
55    from practicals.practical_1.retail_sales
56   where product_category in ('Beauty', 'Electronics');
```

No Database selected Settings Share Open in Workspaces

Results Chart

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
2	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
3	5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
4	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
5	8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
6	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
7	12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
8	13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500

Q8

Practical 1

```
56  -----
57
58  where product_category in ('Beauty', 'Electronics');
59
60  --Q8. Display all transactions where the Product Category is not 'Clothing'.
61  --Expected output: All columns
62
63  select *
64    from practicals.practical_1.retail_sales
65   where product_category not in ('Clothing');
```

No Database selected Settings Share Open in Workspaces

Results Chart

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	150
2	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
3	5	2023-05-06	CUST005	Male	30	Beauty	2	50	100
4	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
5	8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
6	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600

Q9

The screenshot shows a database query editor interface with the title "Practical 1". The top bar includes "ACCOUNTADMIN", "COMPUTE_WH (X-Small)", "Share", and "Open in Workspaces". The code area contains the following SQL query:

```
72
73
74
75 --Q9. Display all transactions where the Quantity is greater than or equal to 3.
76 --Expected output: All columns
77
78 select *
79   from practicals.practical_1.retail_sales
80  where Quantity >= 3 ;
81
82
83
84
```

The results section displays a table with the following data:

# TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	# AGE	PRODUCT_CATEGORY	# QUANTITY	# PRICE_PER_UNIT	# TOTAL_AMOUNT	
14	32	2023-01-04	CUST032	Male	30	Beauty	3	30	90
15	34	2023-12-24	CUST034	Female	51	Clothing	3	50	150
16	35	2023-08-05	CUST035	Female	58	Beauty	3	300	900
17	36	2023-06-24	CUST036	Male	52	Beauty	3	300	900
18	37	2023-05-23	CUST037	Female	18	Beauty	3	25	75
19	38	2023-03-21	CUST038	Male	38	Beauty	4	50	200

Q10

The screenshot shows a database query editor interface with the title "Practical 1". The top bar includes "ACCOUNTADMIN", "COMPUTE_WH (X-Small)", "Share", and "Open in Workspaces". The code area contains the following SQL query:

```
81
82
83
84
85 --Q10. Count the total number of transactions.
86 --Expected output: Total_Transactions
87
88 select count(*) as Total_Transactions
89   from practicals.practical_1.retail_sales;
90
91
92
93
```

The results section displays a table with the following data:

TOTAL_TRANSACTIONS
1
1000

Q11

The screenshot shows a database query interface with the following details:

- Practical 1** tab is selected.
- ACCOUNTADMIN** user is connected to **COMPUTE_WH (X-Small)**.
- No Database selected** and **Settings** dropdowns are visible.
- Share** and **Play** buttons are at the top right.
- Open in Workspaces** and a search icon are also present.
- Code Area:**

```
90  
91  
92  
93  
94 --Q11. Find the average Age of customers.  
95 --Expected output: Average_Age  
96  
97 select AVG(age) as Average_Age  
98   from practicals.practical_1.retail_sales;  
99  
100  
101  
102  
103
```
- Results Tab:** Selected.
- Output:**

#	AVERAGE_AGE
1	41.392000
- Query Details:**
 - Query duration: 64ms
 - Rows: 1
 - Query ID: 01bfc1a3-000c-b0ed-0...

Q12

The screenshot shows a database query interface with the following details:

- Practical 1** tab is selected.
- ACCOUNTADMIN** user is connected to **COMPUTE_WH (X-Small)**.
- No Database selected** and **Settings** dropdowns are visible.
- Share** and **Play** buttons are at the top right.
- Open in Workspaces** and a search icon are also present.
- Code Area:**

```
99  
100  
101  
102  
103 --Q12. Find the total quantity of products sold.  
104 --Expected output: Total_Quantity  
105  
106 select sum(quantity) as Total_Quantity  
107   from practicals.practical_1.retail_sales;  
108  
109  
110  
111 --Q13. Find the maximum Total Amount spent in a single transaction.
```
- Results Tab:** Selected.
- Output:**

#	TOTAL_QUANTITY
1	2514
- Query Details:**
 - Query duration: 64ms
 - Rows: 1
 - Query ID: 01bfc1a3-000c-b0ed-0...

Q13

The screenshot shows a database query editor interface. The top bar displays "Practical 1", "No Database selected", "Settings", "ACCOUNTADMIN", "COMPUTE_WH (X-Small)", "Share", and "Open in Workspaces". The code area contains the following SQL query:

```
108  
109  
110  
111 --Q13. Find the maximum Total Amount spent in a single transaction.  
112 --Expected output: Max_Total_Amount  
113  
114 select max(total_amount) as Max_Total_Amount  
115     from practicals.practical_1.retail_sales;  
116  
117  
118  
119  
120 --Q14. Find the minimum Price per Unit in the dataset.  
121 --Expected output: Min_Price_per_Unit  
122  
123
```

The results tab is active, showing a single row with the header "# MAX_TOTAL_AMOUNT" and a value of 2000. To the right, the "Query Details" panel shows a duration of 28ms, 1 row, and a query ID of 01bfc1a8-000c-b0ed-0... .

Q14

The screenshot shows a database query editor interface. The top bar displays "Practical 1", "No Database selected", "Settings", "ACCOUNTADMIN", "COMPUTE_WH (X-Small)", "Share", and "Open in Workspaces". The code area contains the following SQL query:

```
114     select max(total_amount) as Max_Total_Amount  
115         from practicals.practical_1.retail_sales;  
116  
117  
118  
119  
120 --Q14. Find the minimum Price per Unit in the dataset.  
121 --Expected output: Min_Price_per_Unit  
122  
123 select min(price_per_unit) as Min_Price_per_Unit  
124     from practicals.practical_1.retail_sales;  
125  
126
```

The results tab is active, showing a single row with the header "# MIN_PRICE_PER_UNIT" and a value of 25. To the right, the "Query Details" panel shows a duration of 31ms, 1 row, and a query ID of 01bfc1aa-000c-b0ed-0... .

Q15

The screenshot shows a database query interface with the following details:

Practical 1 tab is selected.

No Database selected.

Query ID: 01bfcae-000c-b0ed-0...

Query duration: 437ms

Rows: 3

Query Details:

```
--Q15. Find the number of transactions per Product Category.  
--Expected output: Product Category, Transaction_Count  
  
select product_category,  
       count(*) as Transaction_Count  
  from practicals.practical_1.retail_sales  
 group by product_category;
```

Results tab is selected.

PRODUCT_CATEGORY	TRANSACTION_COUNT
Clothing	351
Beauty	307
Electronics	342

Q16

The screenshot shows a database query interface with the following details:

Practical 1 tab is selected.

No Database selected.

Query ID: 01bfc1b3-000c-b0ed-0...

Query duration: 85ms

Rows: 2

Query Details:

```
--Q16. Find the total revenue (Total_Amount) per gender.  
--Expected output: Gender, Total_Revenue  
  
select gender,  
       sum(total_amount) as Total_Revenue  
  from practicals.practical_1.retail_sales  
 group by gender;
```

Results tab is selected.

GENDER	TOTAL_REVENUE
Male	223160
Female	232840

Q17

The screenshot shows a database query interface with the following details:

Query:

```
151
152 --Q17. Find the average Price per Unit per product category.
153 --Expected output: Product Category, Average_Price
154
155 select product_category,
156      avg(price_per_unit) as Average_Price
157  from practicals.practical_1.retail_sales
158 group by product_category;
```

Results:

PRODUCT_CATEGORY	AVERAGE_PRICE
Beauty	184.055375
Clothing	174.287749
Electronics	181.900585

Query Details:

- Query duration: 78ms
- Rows: 3
- Query ID: 01bfc1b5-000c-b0ed-0...

Q18

The screenshot shows a database query interface with the following details:

Query:

```
159
160
161
162 --Q18. Find the total revenue per product category where total revenue is greater than 10,000.
163 --Expected output: Product Category, Total_Revenue
164
165 select product_category,
166      sum(total_amount) as Total_Revenue
167  from practicals.practical_1.retail_sales
168 group by product_category
169 having sum(total_amount) > 10000;
```

Results:

PRODUCT_CATEGORY	TOTAL_REVENUE
Beauty	143515
Clothing	155580
Electronics	156905

Query Details:

- Query duration: 85ms
- Rows: 3
- Query ID: 01bfc1bd-000c-b0ed-0...

Q19

Practical 1

No Database selected Settings

Open in Workspaces

```
171  
172  
173 --Q19. Find the average quantity per product category where the average is more than 2.  
174 --Expected output: Product Category, Average_Quantity  
175  
176 select product_category,  
177     avg(quantity) as Average_Quantity  
178     from practicals.practical_1.retail_sales  
179     group by product_category  
180     having avg(quantity) >= 2;  
181  
182  
183
```

Results Chart

PRODUCT_CATEGORY	AVERAGE_QUANTITY
1 Beauty	2.511401
2 Clothing	2.547009
3 Electronics	2.482456

Query Details

Query duration 89ms

Rows 3

Query ID 01bfc1c1-000c-b0ed-00...

Show more

Q20

Practical 1

No Database selected Settings

Open in Workspaces

```
--Q20. Display a column called Spending_Level that shows 'High' if Total Amount > 1000,  
--otherwise 'Low'.  
--Expected output: Transaction ID, Total Amount, Spending_Level  
  
select transaction_id,  
       total_amount,  
         
       case  
       when total_amount > 1000 then 'High'  
       else 'Low'  
       end as Spending_Level  
from practicals.practical_1.retail_sales;
```

Results Chart

# TRANSACTION_ID	# TOTAL_AMOUNT	▲ SPENDING_LEVEL
11	11	100 low
12	12	75 low
13	13	1500 High
14	14	120 low
15	15	2000 High
16	16	1500 High

Q21

The screenshot shows a Snowflake SQL editor interface. The top bar indicates the workspace is 'Practical 1' and the compute type is 'COMPUTE_WH (X-Small)'. The code area contains a query for question 21:

```
200 --Q21. Display a new column called Age_Group that labels customers as:
201 --- 'Youth' if Age < 30
202 --- 'Adult' if Age is between 30 and 59
203 --- 'Senior' if Age >= 60
204 --Expected output: Customer ID, Age, Age_Group
205
206 select customer_id,
207     age,
208     case
209         when age < 30 then 'Youth'
210         when age between 30 and 59 then 'Adult'
211         when age >= 60 then 'Senior'
212     end as Age_Group
213     from practicals.practical_1.retail_sales;
214
```

The results tab shows the output of the query:

CUSTOMER_ID	AGE	AGE_GROUP
CUST007	46	Adult
CUST008	30	Adult
CUST009	63	Senior
CUST010	52	Adult
CUST011	23	Youth

Whole code

--BrightLight Data Analytics Coding Practical

--Practical 1: SQL Fundamentals (Snowflake-Basic SQL Syntax)

--

____QUESTIONS_____

--1. SELECT Statement

--Q1. Display all columns for all transactions.

---Expected output: All columns

```
select *  
from practicals.practical_1.retail_sales;
```

--Q2. Display only the Transaction ID, Date, and Customer ID for all records.

--Expected output: Transaction ID, Date, Customer ID

```
select transaction_id,  
       date,  
       customer_id  
  from practicals.practical_1.retail_sales;
```

--Q3. Display all the distinct product categories in the dataset.

--Expected output: Product Category

```
select distinct(PRODUCT_CATEGORY) as Product_Category  
  from practicals.practical_1.retail_sales;
```

--Q4. Display all the distinct gender values in the dataset.

--Expected output: Gender

```
select distinct(Gender) as GENDER  
from practicals.practical_1.retail_sales;
```

--Q5. Display all transactions where the Age is greater than 40.

--Expected output: All columns

```
select *  
from practicals.practical_1.retail_sales  
where age > 40;
```

--Q6. Display all transactions where the Price per Unit is between 100 and 500.

--Expected output: All columns

```
select *  
from practicals.practical_1.retail_sales  
where price_per_unit between 100 and 500;
```


--Q7. Display all transactions where the Product Category is either 'Beauty' or
--'Electronics'.

```
select *  
from practicals.practical_1.retail_sales  
where product_category in ('Beauty', 'Electronics');
```


--Q8. Display all transactions where the Product Category is not 'Clothing'.
--Expected output: All columns

```
select *  
from practicals.practical_1.retail_sales  
where product_category not in ('Clothing');
```


--Q9. Display all transactions where the Quantity is greater than or equal to 3.
--Expected output: All columns

```
select *
```

```
from practicals.practical_1.retail_sales  
where Quantity >= 3 ;
```

--Q10. Count the total number of transactions.

--Expected output: Total_Transactions

```
select count(*) as Total_Transactions  
from practicals.practical_1.retail_sales;
```

--Q11. Find the average Age of customers.

--Expected output: Average_Age

```
select AVG(age) as Average_Age  
from practicals.practical_1.retail_sales;
```

--Q12. Find the total quantity of products sold.

--Expected output: Total_Quantity

```
select sum(quantity) as Total_Quantity  
from practicals.practical_1.retail_sales;
```

--Q13. Find the maximum Total Amount spent in a single transaction.

--Expected output: Max_Total_Amount

```
select max(total_amount) as Max_Total_Amount  
from practicals.practical_1.retail_sales;
```

--Q14. Find the minimum Price per Unit in the dataset.

--Expected output: Min_Price_per_Unit

```
select min(price_per_unit) as Min_Price_per_Unit  
from practicals.practical_1.retail_sales;
```

--Q15. Find the number of transactions per Product Category.

--Expected output: Product Category, Transaction_Count

```
select product_category,  
       count(*) as Transaction_Count  
  from practicals.practical_1.retail_sales  
 group by product_category;
```

--Q16. Find the total revenue (Total Amount) per gender.

--Expected output: Gender, Total_Revenue

```
select gender,  
       sum(total_amount) as Total_Revenue  
  from practicals.practical_1.retail_sales  
 group by gender;
```

--Q17. Find the average Price per Unit per product category.

--Expected output: Product Category, Average_Price

```
select product_category,  
       avg(price_per_unit) as Average_Price  
  from practicals.practical_1.retail_sales  
group by product_category;
```

--Q18. Find the total revenue per product category where total revenue is greater than 10,000.

--Expected output: Product Category, Total_Revenue

```
select product_category,  
       sum(total_amount) as Total_Revenue  
  from practicals.practical_1.retail_sales  
group by product_category  
having sum(total_amount) > 10000;
```

--Q19. Find the average quantity per product category where the average is more than 2.

--Expected output: Product Category, Average_Quantity

```
select product_category,  
       avg(quantity) as Average_Quantity  
    from practicals.practical_1.retail_sales  
   group by product_category  
having avg(quantity) >= 2;
```

--Q20. Display a column called Spending_Level that shows 'High' if Total Amount > 1000,
--otherwise 'Low'.
--Expected output: Transaction ID, Total Amount, Spending_Level

```
select transaction_id,  
       total_amount,  
       case  
         when total_amount > 1000 then 'High'  
         else 'low'  
       end as Spending_Level  
    from practicals.practical_1.retail_sales;
```

--Q21. Display a new column called Age_Group that labels customers as:

--• 'Youth' if Age < 30

--• 'Adult' if Age is between 30 and 59

--• 'Senior' if Age >= 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'  
         when age >= 60 then 'Senior'  
       end as Age_Group  
     from practicals.practical_1.retail_sales;
```

