

# PRACTICAL EXERCISE 1

**SQL ON SNOWFLAKES**

**Q1.** Display all columns for all transactions. *Expected output:* All columns

The screenshot shows the Snowflake UI interface. On the left is a sidebar with icons for databases, worksheets, and various system status indicators. The main area has tabs for different dates: 2025-04-15 10:59pm, 2025-04-16 12:20am, 2025-04-17 7:00pm, 2025-04-17 7:24pm, 2025-04-17 7:31pm, and 2025-04-18 5:50pm. The current tab is 2025-04-18 5:50pm. At the top right, it shows the user ACCOUNTADMIN and the compute tier COMPUTE\_WH (X-Small), along with share and code version controls. The database selected is SALES.PUBLIC. The query editor contains the following SQL:

```
SELECT*
FROM "SALES".PUBLIC.RETAIL_SALES;
```

The results pane shows a table with the following data:

	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT
1	1	2023-11-24	CUST001	Male	34	Beauty
2	2	2023-02-27	CUST002	Female	26	Clothing
3	3	2023-01-13	CUST003	Male	50	Electronic
4	4	2023-05-21	CUST004	Male	37	Clothing
5	5	2023-05-06	CUST005	Male	30	Beauty
6	6	2023-04-25	CUST006	Female	45	Beauty
7	7	2023-03-13	CUST007	Male	46	Clothing
8	8	2023-02-22	CUST008	Male	30	Electronic

On the right side, there is a 'Query Details' panel showing the duration (23ms), rows (1K), and query ID (01bbccc9-0000-f448-0...). Below that is a histogram for the TRANSACTION\_ID column.

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

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

This screenshot is identical to the one above, showing the same date tabs, user context, and database selection. The query editor contains the following SQL:

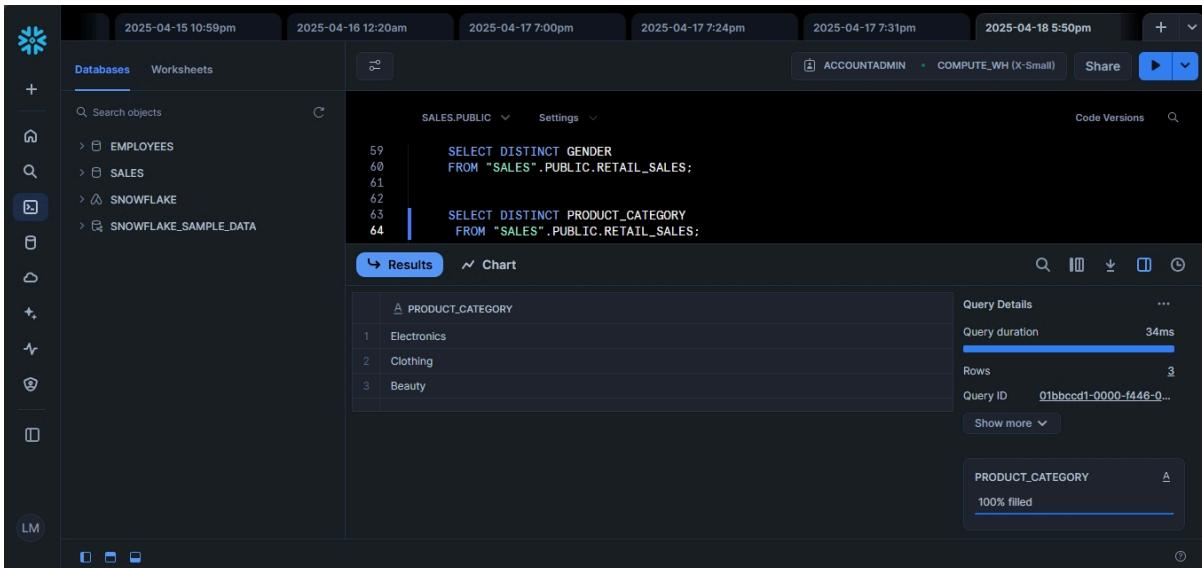
```
SELECT TRANSACTION_ID, DATE, CUSTOMER_ID
FROM "SALES".PUBLIC.RETAIL_SALES;
```

The results pane shows a table with the following data:

	TRANSACTION_ID	DATE	CUSTOMER_ID
1		1	CUST001
2		2	CUST002
3		3	CUST003
4		4	CUST004
5		5	CUST005
6		6	CUST006
7		7	CUST007
8		8	CUST008
9		9	CUST009

The 'Query Details' panel on the right shows a duration of 570ms, 1K rows, and the same query ID. A histogram for the TRANSACTION\_ID column is also present.

**Q3.** Display all the distinct product categories in the dataset. *Expected output:* Product Category



The screenshot shows the Snowflake interface with a dark theme. On the left, there's a sidebar with icons for databases, worksheets, and various system status indicators. The main area has tabs for 'Databases' and 'Worksheets'. A search bar is at the top. Below it, a code editor window displays two queries:

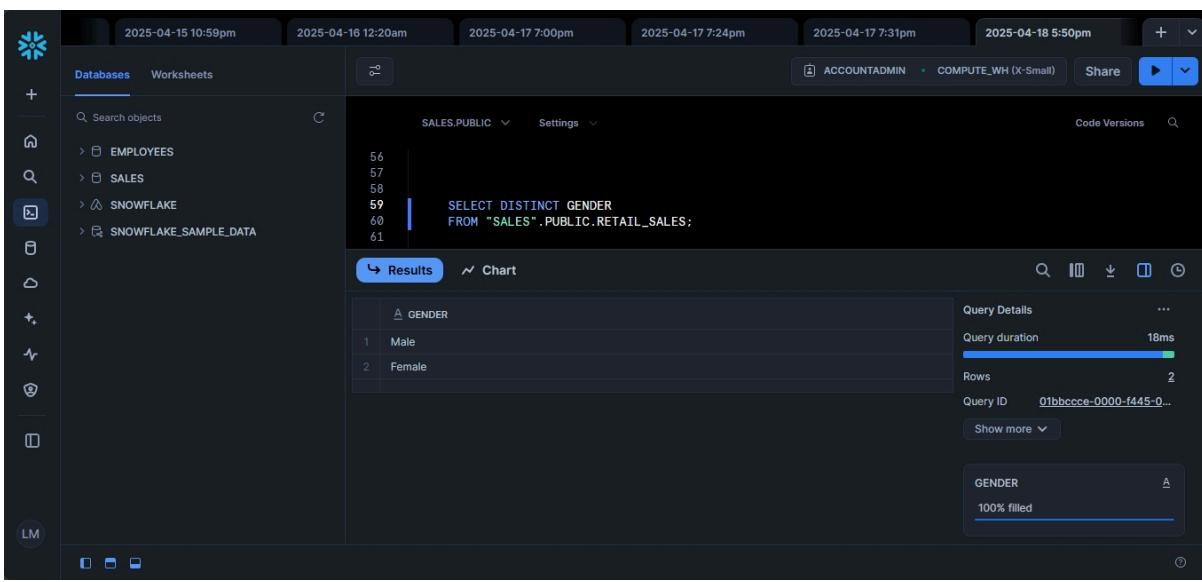
```
SALES.PUBLIC
SELECT DISTINCT GENDER
FROM "SALES".PUBLIC.RETAIL_SALES;

SELECT DISTINCT PRODUCT_CATEGORY
FROM "SALES".PUBLIC.RETAIL_SALES;
```

The second query is currently selected. The results pane shows a table with one column 'PRODUCT\_CATEGORY' containing three rows: Electronics, Clothing, and Beauty. To the right of the results, there's a 'Query Details' panel showing duration (34ms), rows (3), and a query ID. A progress bar indicates the results are 100% filled.

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

*Expected output:* Gender



This screenshot is similar to the previous one, showing the same interface and sidebar. The code editor contains the same two queries, but the second one is selected. The results pane shows a table with one column 'GENDER' containing two rows: Male and Female. The 'Query Details' panel to the right shows a duration of 18ms, 2 rows, and a query ID. A progress bar indicates the results are 100% filled.

**Q5.** Display all transactions where the Age is greater than 40. *Expected output:* All columns

The screenshot shows the Snowflake UI interface. On the left is a sidebar with navigation icons and a search bar. The main area has tabs for 'Databases' and 'Worksheets'. A timeline at the top shows dates from April 15 to April 18. The current worksheet is set to 'SALES.PUBLIC'. The code editor contains the following SQL query:

```
SELECT *
FROM "SALES".PUBLIC.RETAIL_SALES
WHERE AGE > 40;
```

The results table shows 534 rows. The columns are: TRANSACTION\_ID, DATE, CUSTOMER\_ID, GENDER, AGE, and PRODUCT. The data includes various transactions from January to April, with ages ranging from 42 to 63. The right panel displays 'Query Details' with a duration of 51ms and 534 rows, and a histogram for TRANSACTION\_ID.

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT
1	3	2023-01-13	CUST003	Male	50	Electronics
2	6	2023-04-25	CUST006	Female	45	Beauty
3	7	2023-03-13	CUST007	Male	46	Clothing
4	9	2023-12-13	CUST009	Male	63	Electronics
5	10	2023-10-07	CUST010	Female	52	Clothing
6	14	2023-01-17	CUST014	Male	64	Clothing
7	15	2023-01-16	CUST015	Female	42	Electronics
8	18	2023-04-30	CUST018	Female	47	Electronics

**Q6.** Display all transactions where the Price per Unit is between 100 and 500. *Expected output:* All columns

The screenshot shows the Snowflake UI interface. The layout is identical to the previous one, with a sidebar, timeline, and 'SALES.PUBLIC' database selected. The code editor contains the following SQL query:

```
SELECT *
FROM "SALES".PUBLIC.RETAIL_SALES
WHERE PRICE_PER_UNIT BETWEEN '100' AND '500';
```

The results table shows 396 rows. The columns are: AGE, PRODUCT\_CATEGORY, QUANTITY, PRICE\_PER\_UNIT, and TOTAL\_AMOUNT. The data includes various transactions with prices between 100 and 500. The right panel displays 'Query Details' with a duration of 62ms and 396 rows, and a histogram for TRANSACTION\_ID.

#	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	26	Clothing	2	500	1000
2	37	Clothing	1	500	500
3	63	Electronics	2	300	600
4	22	Electronics	3	500	1500
5	42	Electronics	4	500	2000
6	19	Clothing	3	500	1500
7	22	Clothing	3	300	900
8	50	Beauty	1	500	500

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

*Expected output:* All columns

Screenshot of the Snowflake interface showing a query results page. The sidebar shows databases like EMPLOYEES, SALES, SNOWFLAKE, and SNOWFLAKE\_SAMPLE\_DATA. The main area displays a query in the code editor and its results in a table. The table has columns: TRANSACTION\_ID, DATE, CUSTOMER\_ID, GENDER, AGE, and PRODUCT. The results show transactions for Beauty and Electronics categories. A chart and query details are also visible.

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

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

Screenshot of the Snowflake interface showing a query results page. The sidebar shows databases like EMPLOYEES, SALES, SNOWFLAKE, and SNOWFLAKE\_SAMPLE\_DATA. The main area displays a query in the code editor and its results in a table. The table has columns: TRANSACTION\_ID, DATE, CUSTOMER\_ID, GENDER, AGE, and PRODUCT. The results show transactions for Beauty and Electronics categories, excluding Clothing. A chart and query details are also visible.

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

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

The screenshot shows the Snowflake UI interface. On the left is a sidebar with icons for databases, worksheets, and various system objects. The main area has tabs for different dates: 2025-04-15 10:59pm, 2025-04-16 12:20am, 2025-04-17 7:00pm, 2025-04-17 7:24pm, 2025-04-17 7:31pm, and 2025-04-18 5:50pm. The current tab is 2025-04-17 7:31pm. At the top right, it shows the user ACCOUNTADMIN and the compute type COMPUTE\_WH (x-Small). Below the tabs, there's a search bar and a code editor with the following SQL query:

```
116 WHERE PRODUCT_CATEGORY not in ('Clothing');
117
118
119
120
121
SELECT *
FROM "SALES".PUBLIC.RETAIL_SALES
WHERE QUANTITY >= ('3');
```

The results pane shows a table with columns: FOMER\_ID, GENDER, AGE, PRODUCT\_CATEGORY, QUANTITY, and PRICE\_PER\_UNIT. The data is as follows:

FOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT
1 1	Male	34	Beauty	3	
2 8	Male	30	Electronics	4	
3 10	Female	52	Clothing	4	
4 12	Male	35	Beauty	3	
5 13	Male	22	Electronics	3	
6 14	Male	64	Clothing	4	
7 15	Female	42	Electronics	4	
8 16	Male	19	Clothing	3	

On the right side, there are sections for 'Query Details' (duration 60ms, rows 504, query ID 01bbccf1-0000-efae-00...), a histogram for TRANSACTION\_ID, and other UI elements like a search bar and a refresh button.

**Q10.** Count the total number of transactions.

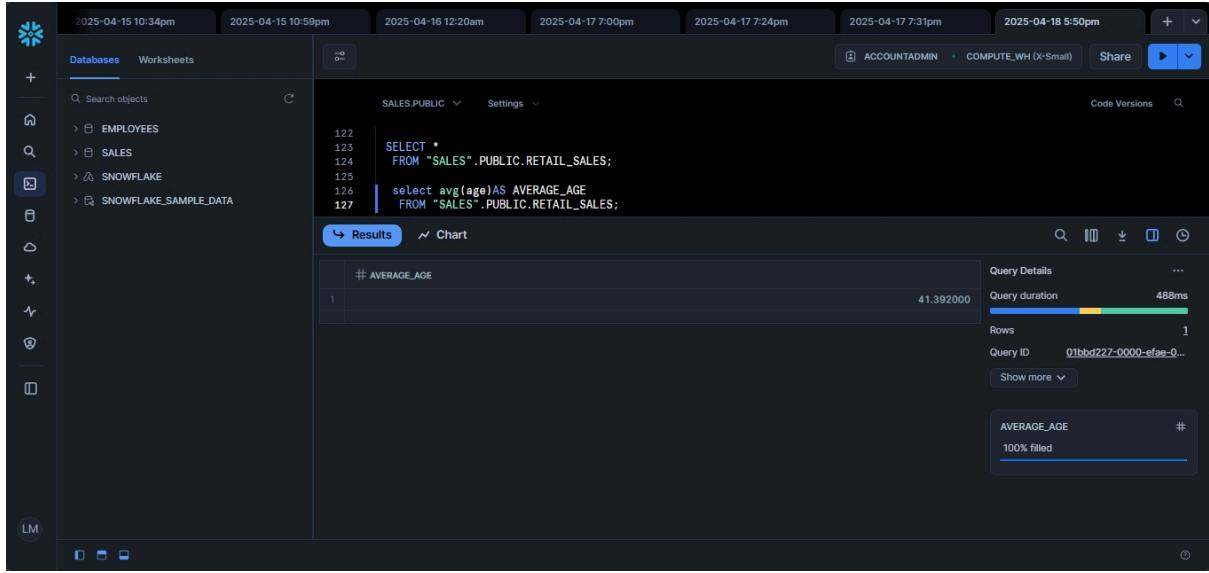
*Expected output:* Total\_Transactions

This screenshot shows the same Snowflake UI setup as the previous one. The current tab is 2025-04-17 7:31pm. The code editor contains the following SQL query:

```
62 FROM "SALES".PUBLIC.RETAIL_SALES
63 WHERE Quantity >= ('3');
64
65
66 select count(Transaction_id) AS Total_Transaction
67 from "SALES".PUBLIC.RETAIL_SALES;
```

The results pane shows a single row with the column TOTAL\_TRANSACTION and the value 1000. The 'Query Details' section on the right shows a duration of 18ms, 1 row, and a query ID of 01bbcde8-0000-f405-00... The histogram for TOTAL\_TRANSACTION is 100% filled.

**Q11.** Find the average Age of customers. *Expected output:* Average\_Age



Snowflake Worksheet interface showing a query result for the average age of customers.

Query:

```
SELECT *  
FROM "SALES".PUBLIC.RETAIL_SALES;  
  
select avg(age)AS AVERAGE_AGE  
FROM "SALES".PUBLIC.RETAIL_SALES;
```

Results:

AVERAGE_AGE
41.392000

Query Details:

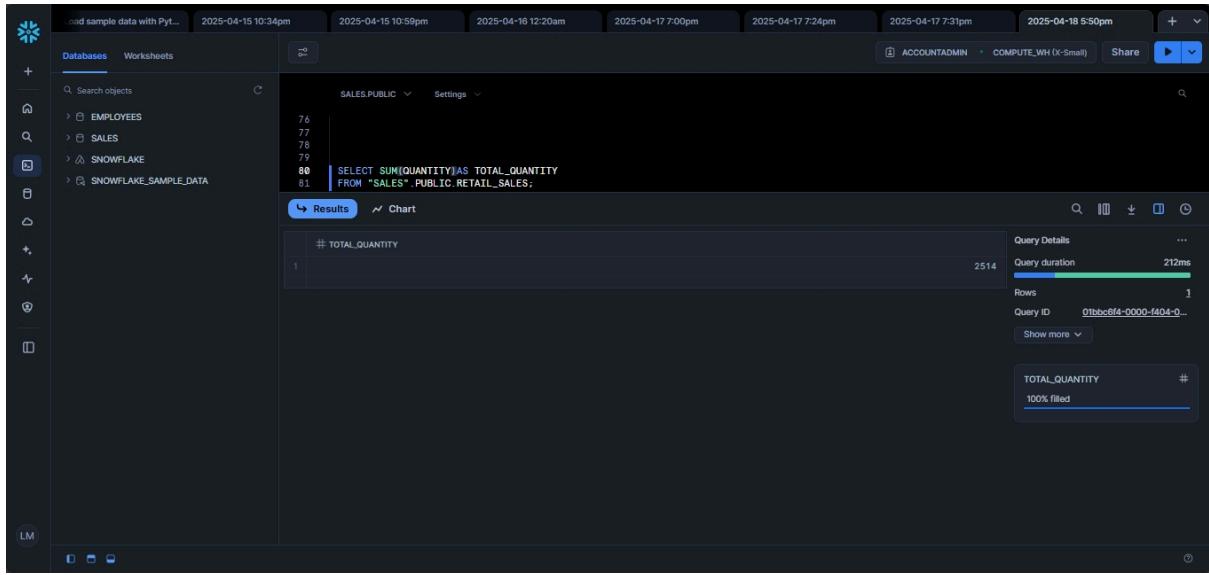
- Query duration: 488ms
- Rows: 1
- Query ID: 01bbd227-0000-efae-0...

Chart:

AVERAGE\_AGE

100% filled

**Q12.** Find the total quantity of products sold. *Expected output:* Total\_Quantity



Snowflake Worksheet interface showing a query result for the total quantity of products sold.

Query:

```
SELECT SUM(QUANTITY)AS TOTAL_QUANTITY  
FROM "SALES".PUBLIC.RETAIL_SALES;
```

Results:

TOTAL_QUANTITY
2514

Query Details:

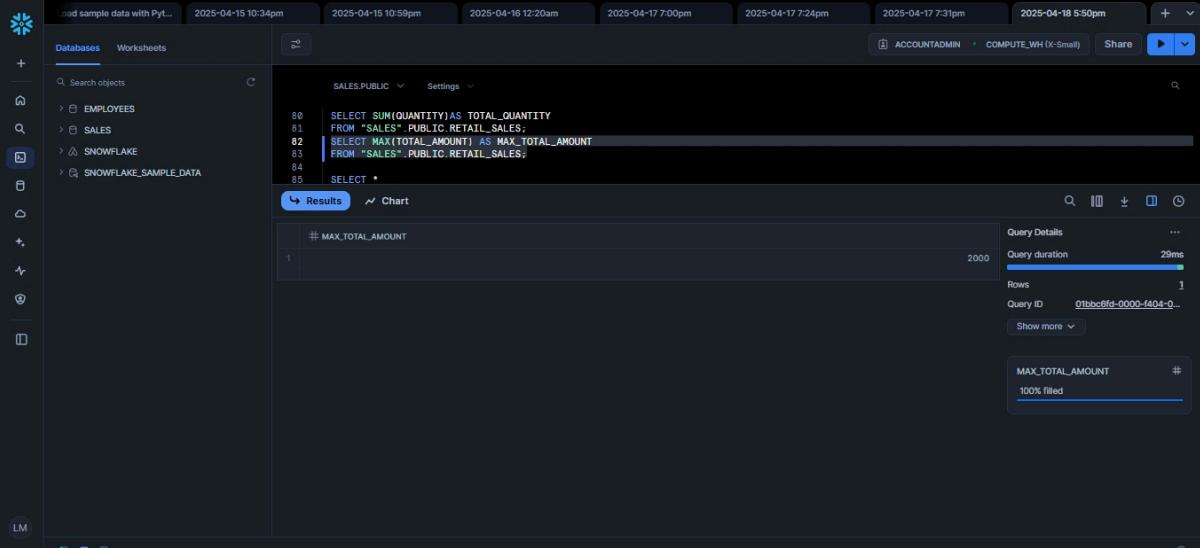
- Query duration: 212ms
- Rows: 1
- Query ID: 01bbc6f4-0000-f404-0...

Chart:

TOTAL\_QUANTITY

100% filled

**Q13.** Find the maximum Total Amount spent in a single transaction. *Expected output:* Max\_Total\_Amount



The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for databases, worksheets, and other tools. The main area has tabs for 'Databases' and 'Worksheets'. A search bar at the top says 'Search objects'. Below it, a tree view shows 'EMPLOYEES', 'SALES', 'SNOWFLAKE', and 'SNOWFLAKE\_SAMPLE\_DATA'. The 'Worksheets' tab is selected. In the center, a code editor window titled 'SALES.PUBLIC' contains the following SQL query:

```
80  SELECT SUM(QANTITY)AS TOTAL_QUANTITY  
81  FROM "SALES".PUBLIC.RETAIL_SALES;  
82  SELECT MAX(TOTAL_AMOUNT) AS MAX_TOTAL_AMOUNT  
83  FROM "SALES".PUBLIC.RETAIL_SALES;  
84  
85  SELECT *
```

Below the code editor is a results table with one row:

#	MAX_TOTAL_AMOUNT
1	2000

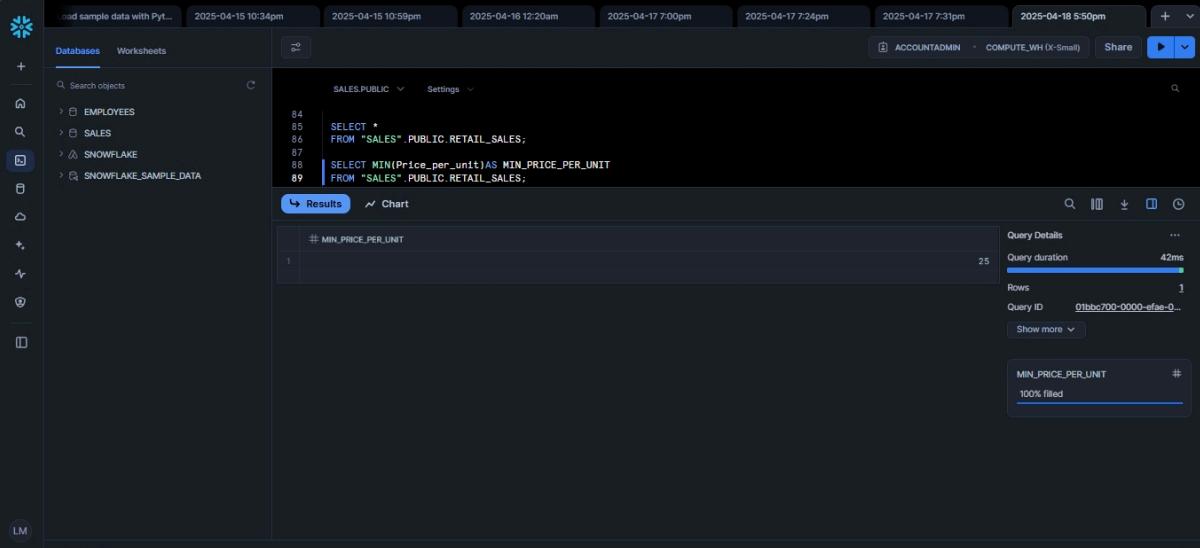
On the right side, there's a 'Query Details' panel showing:

- Query duration: 29ms
- Rows: 1
- Query ID: 01bbc8fd-0000-f404-0...

A progress bar indicates '100% filled'.

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

*Expected output:* Min\_Price\_per\_Unit



The screenshot shows the Snowflake UI interface, similar to the previous one. The sidebar and tree view are identical. The 'Worksheets' tab is selected. The code editor window titled 'SALES.PUBLIC' contains the following SQL query:

```
84  
85  SELECT *  
86  FROM "SALES".PUBLIC.RETAIL_SALES;  
87  
88  SELECT MIN(Price_per_unit)AS MIN_PRICE_PER_UNIT  
89  FROM "SALES".PUBLIC.RETAIL_SALES;
```

The results table shows one row:

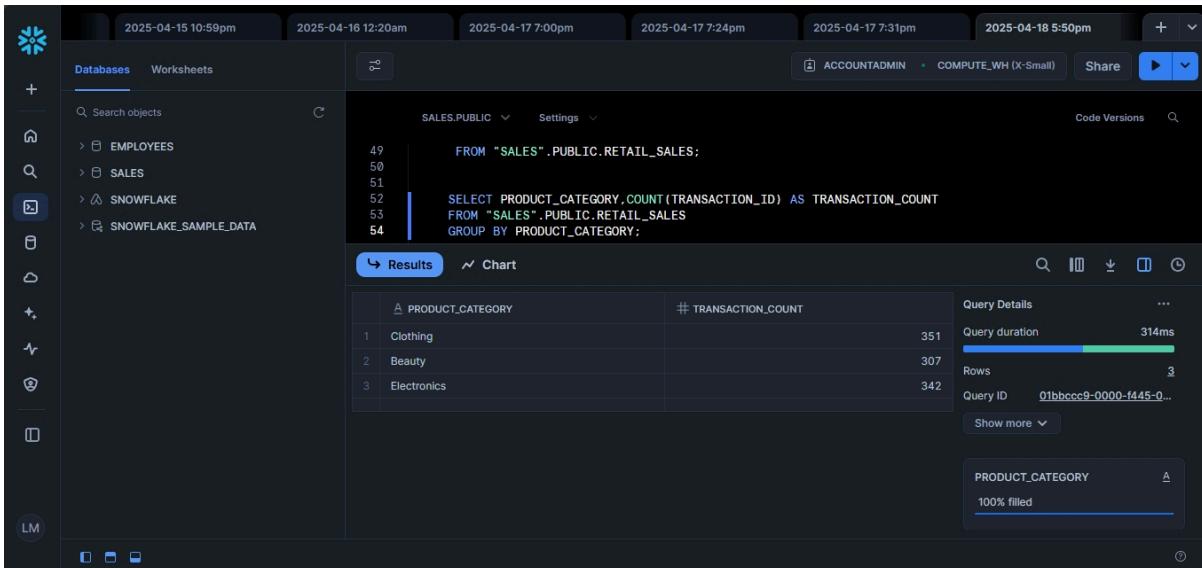
#	MIN_PRICE_PER_UNIT
1	25

The 'Query Details' panel on the right shows:

- Query duration: 42ms
- Rows: 1
- Query ID: 01bbc700-0000-efae-0...

A progress bar indicates '100% filled'.

**Q15.** Find the number of transactions per Product Category. *Expected output:* Product Category, Transaction\_Count



Snowflake Worksheet interface showing a query results page. The sidebar on the left lists databases (EMPLOYEES, SALES, SNOWFLAKE, SNOWFLAKE\_SAMPLE\_DATA) and worksheets. The main area shows a query in the editor and its results in a table.

**Query Editor:**

```
FROM "SALES".PUBLIC.RETAIL_SALES;
SELECT PRODUCT_CATEGORY,COUNT(TRANSACTION_ID) AS TRANSACTION_COUNT
FROM "SALES".PUBLIC.RETAIL_SALES
GROUP BY PRODUCT_CATEGORY;
```

**Results Table:**

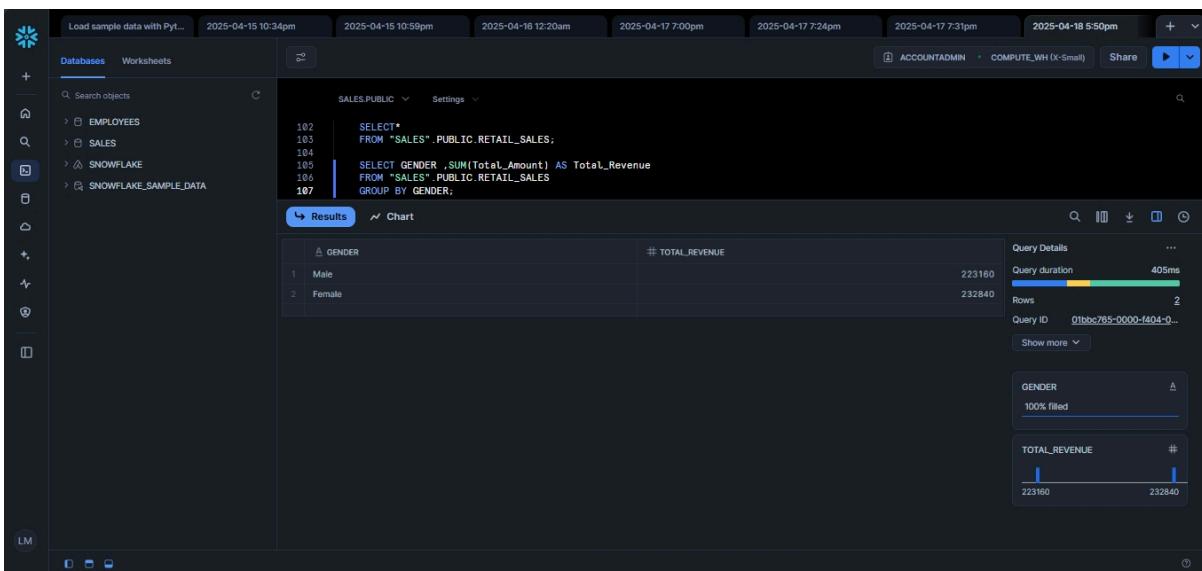
PRODUCT_CATEGORY	TRANSACTION_COUNT
Clothing	351
Beauty	307
Electronics	342

**Query Details:**

- Query duration: 314ms
- Rows: 3
- Query ID: 01bbc765-0000-4445-0...

Chart preview: PRODUCT\_CATEGORY (100% filled)

**Q16.** Find the total revenue (Total Amount) per gender. *Expected output:* Gender, Total\_Revenue



Snowflake Worksheet interface showing a query results page. The sidebar on the left lists databases (EMPLOYEES, SALES, SNOWFLAKE, SNOWFLAKE\_SAMPLE\_DATA) and worksheets. The main area shows a query in the editor and its results in a table.

**Query Editor:**

```
SELECT*
FROM "SALES".PUBLIC.RETAIL_SALES;
SELECT GENDER ,SUM(Total_Amount) AS Total_Revenue
FROM "SALES".PUBLIC.RETAIL_SALES
GROUP BY GENDER;
```

**Results Table:**

GENDER	TOTAL_REVENUE
Male	223160
Female	232840

**Query Details:**

- Query duration: 405ms
- Rows: 2
- Query ID: 01bbc765-0000-4445-0...

Chart preview: GENDER (100% filled)  
TOTAL\_REVENUE (#)

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

*Expected output:* Product Category, Average\_Price

The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for databases, worksheets, and search objects. The main area shows a worksheet titled 'SALES.PUBLIC' with the following SQL code:

```
108
109
110
111
112
113
114
115
116
117
118
SELECT Product_category,
       AVG (Price_Per_Unit) AS AVERAGE_PRICE
  FROM "SALES".PUBLIC.RETAIL_SALES
 GROUP BY PRODUCT_CATEGORY;
```

The results table shows the average price for three categories:

PRODUCT_CATEGORY	AVERAGE_PRICE
Clothing	174.287749
Beauty	184.055375
Electronics	181.900585

On the right, there are 'Query Details' and two charts. The first chart is for 'PRODUCT\_CATEGORY' with values 100% filled. The second chart is for 'AVERAGE\_PRICE' with values 174.287749 and 184.055375.

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

*Expected output:* Product Category, Total\_Revenue

The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for databases, worksheets, and search objects. The main area shows a worksheet titled 'SALES.PUBLIC' with the following SQL code:

```
113
114
115
116
117
118
SELECT Product_Category, SUM(TOTAL_AMOUNT) AS TOTAL_REVENUE
  FROM "SALES".PUBLIC.RETAIL_SALES
 GROUP BY PRODUCT_CATEGORY
 HAVING TOTAL_REVENUE > 10000;
```

The results table shows the total revenue for three categories:

PRODUCT_CATEGORY	TOTAL_REVENUE
Clothing	155580
Beauty	143515
Electronics	156905

On the right, there are 'Query Details' and two charts. The first chart is for 'PRODUCT\_CATEGORY' with values 100% filled. The second chart is for 'TOTAL\_REVENUE' with values 143515 and 156905.

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

*Expected output:* Product Category, Average\_Price

The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for databases, worksheets, and search objects. The main area shows a worksheet titled 'SALES.PUBLIC' with the following SQL query:

```
108
109
110
111
112
113
114
115
116
117
118
SELECT Product_category,
       AVG (Price_Per_Unit) AS AVERAGE_PRICE
  FROM "SALES".PUBLIC.RETAIL_SALES
 GROUP BY PRODUCT_CATEGORY;
```

The results table shows the average price for three categories:

PRODUCT_CATEGORY	AVERAGE_PRICE
Clothing	174.287749
Beauty	184.055375
Electronics	181.900585

On the right, there are 'Query Details' and two charts. The first chart is for 'PRODUCT\_CATEGORY' with values 100% filled. The second chart is for 'AVERAGE\_PRICE' with values 174.287749 and 184.055375.

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

*Expected output:* Product Category, Total\_Revenue

The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for databases, worksheets, and search objects. The main area shows a worksheet titled 'SALES.PUBLIC' with the following SQL query:

```
113
114
115
116
117
118
SELECT Product_Category, SUM(TOTAL_AMOUNT) AS TOTAL_REVENUE
  FROM "SALES".PUBLIC.RETAIL_SALES
 GROUP BY PRODUCT_CATEGORY
 HAVING TOTAL_REVENUE > 10000;
```

The results table shows the total revenue for three categories:

PRODUCT_CATEGORY	TOTAL_REVENUE
Clothing	155580
Beauty	143515
Electronics	156905

On the right, there are 'Query Details' and two charts. The first chart is for 'PRODUCT\_CATEGORY' with values 100% filled. The second chart is for 'TOTAL\_REVENUE' with values 143515 and 156905.

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

*Expected output:* Product Category, Average\_Quantity

The screenshot shows the Snowflake interface with a query editor and a results viewer. The query is:

```
122
123 | SELECT PRODUCT_CATEGORY,
124 |     AVG(QUANTITY) AS AVERAGE_QUANTITY
125 |     FROM "SALES".PUBLIC.RETAIL_SALES
126 |     GROUP BY PRODUCT_CATEGORY
127 |     HAVING AVERAGE_QUANTITY > 2;
```

The results table has two columns: PRODUCT\_CATEGORY and AVERAGE\_QUANTITY. The data is:

PRODUCT_CATEGORY	AVERAGE_QUANTITY
Clothing	2.547009
Beauty	2.511401
Electronics	2.482456

On the right side, there are 'Query Details' and two charts. The first chart shows 'PRODUCT\_CATEGORY' with 100% filled. The second chart shows 'AVERAGE\_QUANTITY' with values 2.482456 and 2.547009.

**Q20.** Display a column called Spending\_Level that shows 'High' if Total Amount > 1000, otherwise 'Low'.

*Expected output:* Transaction ID, Total Amount, Spending\_Level

The screenshot shows the Snowflake interface with a query editor and a results viewer. The query is:

```
6
7
8 | SELECT TRANSACTION_ID, TOTAL_AMOUNT, CASE WHEN TOTAL_AMOUNT > '1000' THEN 'HIGH'
9 | ELSE 'LOW'
10 | END AS SPENDING_LEVEL
11 | FROM "SALES".PUBLIC.RETAIL_SALES;
```

The results table has three columns: TRANSACTION\_ID, TOTAL\_AMOUNT, and SPENDING\_LEVEL. The data is:

TRANSACTION_ID	TOTAL_AMOUNT	SPENDING_LEVEL
1	150	LOW
2	1000	HIGH
3	30	LOW
4	500	LOW
5	100	LOW
6	30	LOW
7	50	LOW
8	100	LOW
9	600	LOW
10	200	LOW
11	100	LOW
12	75	LOW
13	1500	HIGH
14	120	LOW
15	2000	HIGH
16	1500	HIGH

On the right side, there are 'Query Details' and three charts. The first chart shows 'TRANSACTION\_ID' with value 1000. The second chart shows 'TOTAL\_AMOUNT' with values 25, 100, 200, and 2000. The third chart shows 'SPENDING\_LEVEL' with values LOW and HIGH.

**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

The screenshot shows the Snowflake UI interface. On the left is a sidebar with various icons and a search bar. The main area has a timeline at the top with dates from April 15 to April 18. Below the timeline, there are tabs for 'Databases' and 'Worksheets'. A 'SALES.PUBLIC' database is selected. The code editor contains a query:

```
44  SELECT Customer_id, Age,  
45  CASE  
46  WHEN Age < '30' THEN 'YOUTH'  
47  WHEN Age BETWEEN '30' AND '59' THEN 'Adult'  
48  WHEN Age >= '60' THEN 'SENIOR' END AS AGE_GROUP;  
49  FROM "SALES".PUBLIC.RETAIL_SALES;
```

The results tab is active, showing a table with three columns: CUSTOMER\_ID, AGE, and AGE\_GROUP. The data is as follows:

	CUSTOMER_ID	AGE	AGE_GROUP
1	CUST001	34	Adult
2	CUST002	26	YOUTH
3	CUST003	50	Adult
4	CUST004	37	Adult
5	CUST005	30	Adult
6	CUST006	45	Adult
7	CUST007	46	Adult
8	CUST008	30	Adult
9	CUST009	63	SENIOR

On the right side, there is a 'Query Details' panel showing the duration (114ms), rows (1K), and query ID (01bbccc1-0000-efae-0...). It also indicates that the CUSTOMERID column is 100% filled.