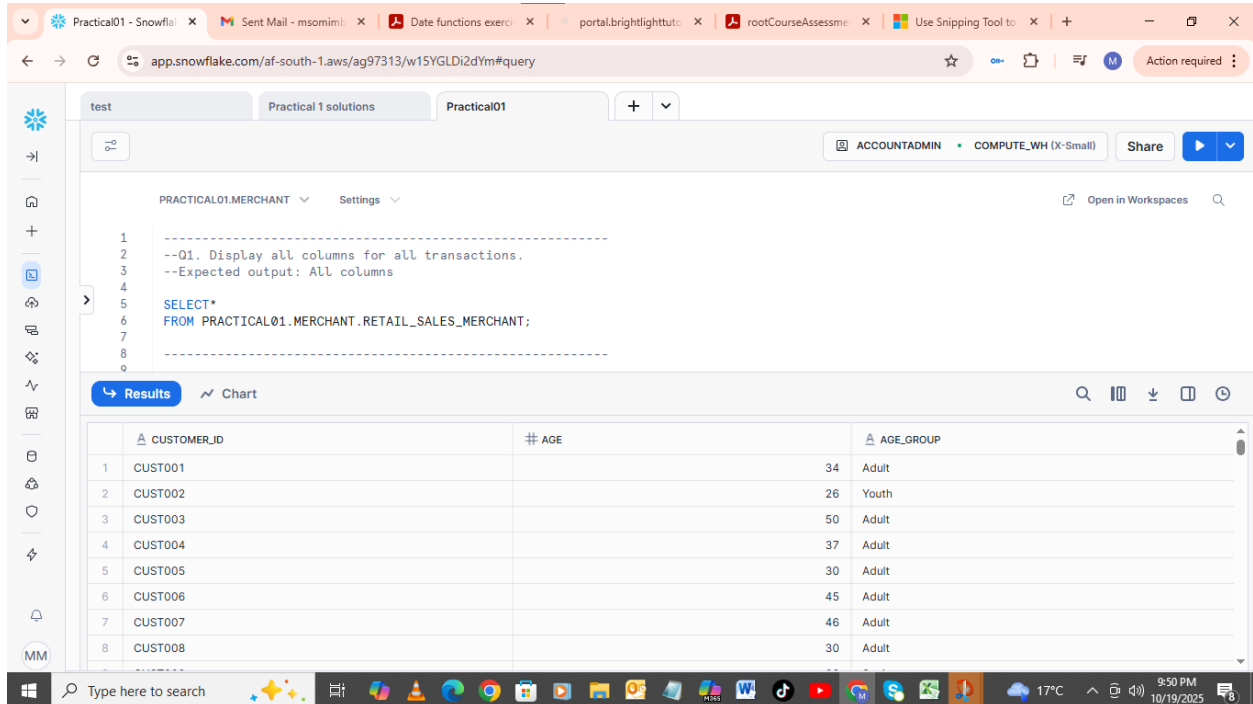


Mbali Zondi

Practical 1: Basic SQL Syntax

Question 1:



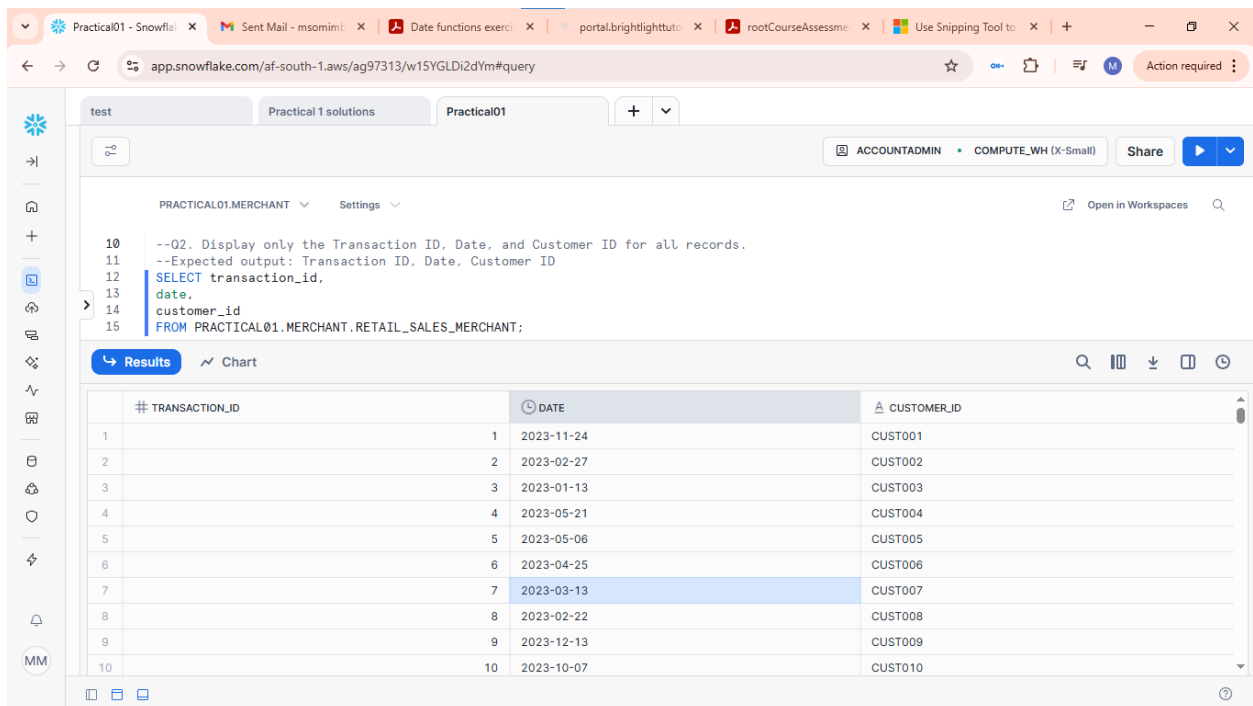
The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
1  -----  
2  --Q1. Display all columns for all transactions.  
3  --Expected output: All columns  
4  
5  SELECT*  
6  FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;  
7  
8  -----  
9
```

The results table shows the following data:

	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

Question 2:



The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
10  
11 --Q2. Display only the Transaction ID, Date, and Customer ID for all records.  
12 --Expected output: Transaction ID, Date, Customer ID  
13 SELECT transaction_id,  
14 date,  
15 customer_id  
16 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The results table shows the following data:

	TRANSACTION_ID	DATE	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
10	10	2023-10-07	CUST010

Question 3:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
17 -----
18
19 --Q3. Display all the distinct product categories in the dataset.
20 --Expected output: Product Category
21 SELECT DISTINCT product_category
22 FROM practical01.merchant.retail_sales_merchant;
```

The query results are displayed in a table with the following data:

	PRODUCT_CATEGORY
1	Clothing
2	Beauty
3	Electronics

Question 4:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
25
26 --Q4. Display all the distinct gender values in the dataset.
27 --Expected output: Gender
28 SELECT DISTINCT gender
29 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
30
```

The query results are displayed in a table with the following data:

	GENDER
1	Male
2	Female

Question 5:

app.snowflake.com/af-south-1.aws/ag97313/w15YGLDi2dYm#query

test Practical 1 solutions Practical01

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

PRACTICAL01.MERCHANT Settings Open in Workspaces

```
33 --Q5. Display all transactions where the Age is greater than 40.
34 --Expected output: All columns
35 SELECT*
36 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT
37 WHERE age> 40;
38
```

Results Chart

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	3	2023-01-13	CUST003	Male	50	Electronics	1	30	30
2	6	2023-04-25	CUST006	Female	45	Beauty	1	30	30
3	7	2023-03-13	CUST007	Male	46	Clothing	2	25	50
4	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
5	10	2023-10-07	CUST010	Female	52	Clothing	4	50	200
6	14	2023-01-17	CUST014	Male	64	Clothing	4	30	120
7	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
8	18	2023-04-30	CUST018	Female	47	Electronics	2	25	50
9	19	2023-09-16	CUST019	Female	62	Clothing	2	25	50
10	21	2023-01-14	CUST021	Female	50	Beauty	1	500	500

Question 6:

app.snowflake.com/af-south-1.aws/ag97313/w15YGLDi2dYm#query

test Practical 1 solutions Practical01

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

PRACTICAL01.MERCHANT Settings Open in Workspaces

```
41 --Q6. Display all transactions where the Price per Unit is between 100 and 500.
42 --Expected output: All columns
43 SELECT*
44 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT
45 WHERE Price_per_unit BETWEEN 100 AND 500;
46
```

Results Chart

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL_AMOUNT
1	2	2023-02-27	CUST002	Female	26	Clothing	2	500	1000
2	4	2023-05-21	CUST004	Male	37	Clothing	1	500	500
3	9	2023-12-13	CUST009	Male	63	Electronics	2	300	600
4	13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
5	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
6	16	2023-02-17	CUST016	Male	19	Clothing	3	500	1500
7	20	2023-11-05	CUST020	Male	22	Clothing	3	300	900
8	21	2023-01-14	CUST021	Female	50	Beauty	1	500	500
9	24	2023-11-29	CUST024	Female	49	Clothing	1	300	300
10	26	2023-10-07	CUST026	Female	28	Electronics	2	500	1000

Question 7:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--Q7. Display all transactions where the Product Category is either 'Beauty' or 'Electronics'.  
--Expected output: All columns  
SELECT*  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
WHERE Product_category IN ('Beauty','Electronics');
```

The results table displays the following data:

#	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
9	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
10	18	2023-04-30	CUST018	Female	47	Electronics	2	25	50

Question 8:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--Q8. Display all transactions where the Product Category is not 'Clothing'.  
--Expected output: All columns  
SELECT*  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
WHERE Product_category != 'Clothing';
```

The results table displays the following data:

#	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

Question 9:

app.snowflake.com/af-south-1.aws/ag97313/w15YGLDi2dYm#query

test Practical 1 solutions Practical01

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

PRACTICAL01.MERCHANT Settings Open in Workspaces

```

65 --Q9. Display all transactions where the Quantity is greater than or equal to 3.
66 --Expected output: All columns
67 SELECT*
68 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT
69 WHERE Quantity >=3;
70

```

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	8	2023-02-22	CUST008	Male	30	Electronics	4	25	100
3	10	2023-10-07	CUST010	Female	52	Clothing	4	50	200
4	12	2023-10-30	CUST012	Male	35	Beauty	3	25	75
5	13	2023-08-05	CUST013	Male	22	Electronics	3	500	1500
6	14	2023-01-17	CUST014	Male	64	Clothing	4	30	120
7	15	2023-01-16	CUST015	Female	42	Electronics	4	500	2000
8	16	2023-02-17	CUST016	Male	19	Clothing	3	500	1500
9	17	2023-04-22	CUST017	Female	27	Clothing	4	25	100
10	20	2023-11-05	CUST020	Male	22	Clothing	3	300	900

Question 10:

app.snowflake.com/af-south-1.aws/ag97313/w15YGLDi2dYm#query

test Practical 1 solutions Practical01

ACCOUNTADMIN COMPUTE_WH (X-Small) Share

PRACTICAL01.MERCHANT Settings Open in Workspaces

```

71 -----
72
73 --Q10. Count the total number of transactions.
74 --Expected output: Total_Transactions
75 SELECT COUNT(*) As Total_Transactions
76 FROM practical01.merchant.retail_sales_merchant;

```

Results Chart

#	TOTAL_TRANSACTIONS
1	1000

Question 11:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--Q11. Find the average Age of customers.  
--Expected output: Average_Age  
SELECT AVG (Age) As AVERAGE_Age  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The query results are displayed in a table with the following data:

# AVERAGE_AGE	
1	41.392000

The interface also shows the user 'ACCOUNTADMIN' and the warehouse 'COMPUTE_WH (X-Small)'.

Question 12:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--Q12. Find the total quantity of products sold.  
--Expected output: Total_Quantity  
SELECT SUM (Quantity) As Total_Quantity  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The query results are displayed in a table with the following data:

# TOTAL_QUANTITY	
1	2514

The interface also shows the user 'ACCOUNTADMIN' and the warehouse 'COMPUTE_WH (X-Small)'.

Question 13:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--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 PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The results pane shows a single row with the value 2000.

#	MAX_TOTAL_AMOUNT
1	2000

Question 14:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--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 PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The results pane shows a single row with the value 25.

#	MIN_PRICE_PER_UNIT
1	25

Question 15:

The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
--Q15. Find the number of transactions per Product Category.  
--Expected output: Product Category, Transaction_Count  
SELECT Product_Category,  
COUNT (*) As Transaction_Count  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
GROUP BY Product_Category;
```

The results table shows the following data:

	PRODUCT_CATEGORY	# TRANSACTION_COUNT
1	Clothing	351
2	Beauty	307
3	Electronics	342

Question 16:

The screenshot shows the Snowflake web interface. The query editor displays the following SQL code:

```
--Q16. Find the total revenue (Total Amount) per gender.  
--Expected output: Gender, Total_Revenue  
SELECT Gender,  
SUM (Total_Amount) As Total_Revenue  
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
GROUP BY Gender;
```

The results table shows the following data:

	GENDER	# TOTAL_REVENUE
1	Male	223160
2	Female	232840

Question 17:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--017. 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 PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
GROUP BY Product_Category;
```

The results table shows the following data:

	PRODUCT_CATEGORY	AVERAGE_PRICE
1	Beauty	184.055375
2	Clothing	174.287749
3	Electronics	181.900585

Question 18:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--018. 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 PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT  
GROUP BY Product_Category  
HAVING SUM(Total_Amount)> 10000;
```

The results table shows the following data:

	PRODUCT_CATEGORY	TOTAL_REVENUE
1	Beauty	143515
2	Clothing	155580
3	Electronics	156905

Question 19:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
--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 PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT
GROUP BY Product_Category
HAVING AVG (Quantity) > 2;
```

The results table shows the following data:

	PRODUCT_CATEGORY	AVERAGE_QUANTITY
1	Beauty	2.511401
2	Clothing	2.547009
3	Electronics	2.482456

Question 20:

The screenshot shows the Snowflake web interface. The query editor contains the following SQL code:

```
SELECT Transaction_ID,
       Total_Amount,
       CASE
         WHEN Total_Amount > 1000 THEN 'High'
         ELSE 'Low'
       END AS Spendind_Level
FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
```

The results table shows the following data:

# TRANSACTION_ID	# TOTAL_AMOUNT	SPENDING_LEVEL
1	150	Low
2	1000	Low
3	30	Low
4	500	Low
5	100	Low
6	30	Low
7	50	Low
8	100	Low

Question 21:

The screenshot shows the Snowflake web interface. The browser address bar displays the URL: `app.snowflake.com/af-south-1.aws/ag97313/w15YGLDi2dYm#query`. The interface includes a sidebar with navigation icons, a top navigation bar with tabs for 'test', 'Practical 1 solutions', and 'Practical01'. The main area shows a SQL query being executed on the 'PRACTICAL01.MERCHANT' database. The query is as follows:

```
175 SELECT Customer_ID,
176        Age,
177        CASE
178          WHEN Age < 30 THEN 'Youth'
179          WHEN Age BETWEEN 30 AND 59 THEN 'Adult'
180          WHEN Age >= 60 THEN 'Senior'
181        END AS Age_Group
182 FROM PRACTICAL01.MERCHANT.RETAIL_SALES_MERCHANT;
183
```

Below the query, the 'Results' tab is active, displaying a table with 8 rows. The table has 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

The bottom of the image shows a Windows taskbar with various application icons, a search bar, and system information including the temperature (17°C) and date/time (9:40 PM, 10/19/2025).