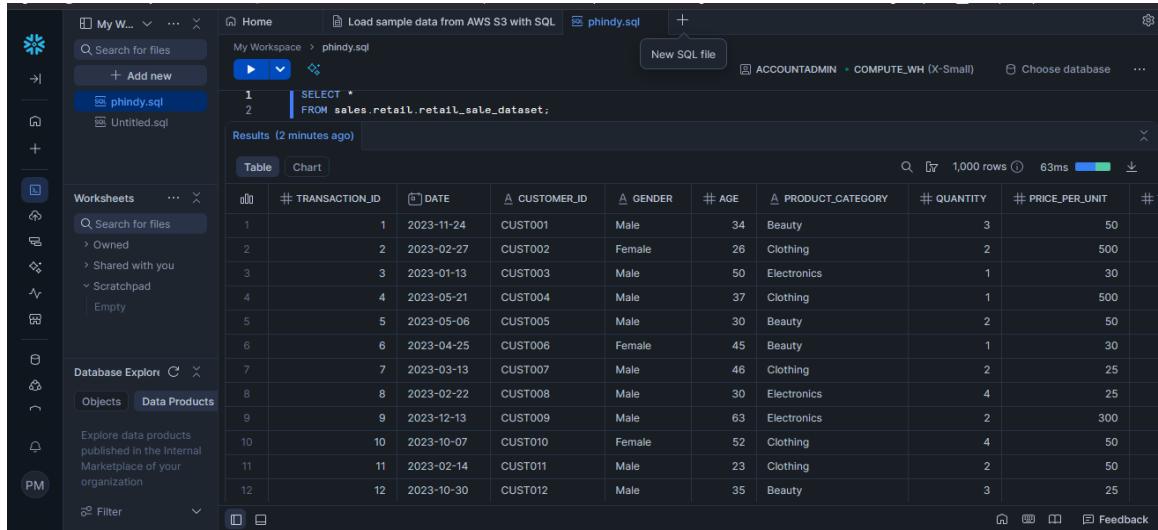


# Phindile Mnisi

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

### 1. SELECT Statement

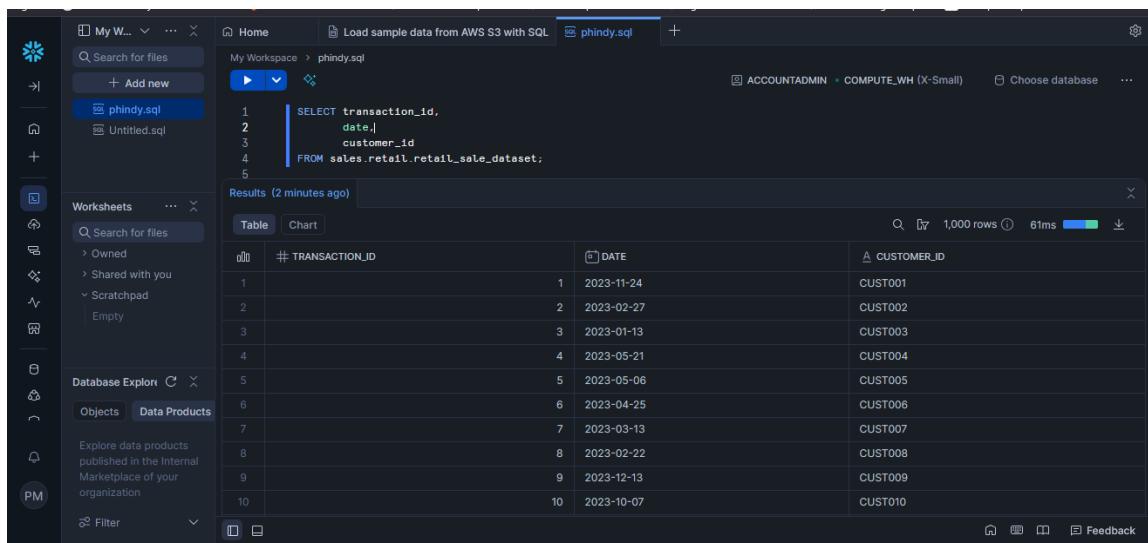


The screenshot shows the Snowflake UI interface. On the left, there's a sidebar with icons for My Workspace, Worksheets, Database Explorer, and PM. The main area has tabs for Home, Load sample data from AWS S3 with SQL, and phindy.sql. The phindy.sql tab is active, displaying the following SQL code:

```
1 | SELECT *
2 | FROM sales.retail.retail_sale_dataset;
```

The Results section shows a table with 12 rows of data. The columns are: # TRANSACTION\_ID, DATE, CUSTOMER\_ID, GENDER, AGE, PRODUCT\_CATEGORY, QUANTITY, PRICE\_PER\_UNIT, and #. The data includes various transaction details like dates (2023-11-24 to 2023-10-30), customer IDs (CUST001 to CUST012), and product categories like Beauty, Clothing, and Electronics.

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	#
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	
2	2	2023-02-27	CUST002	Female	26	Clothing	2	500	
3	3	2023-01-13	CUST003	Male	50	Electronics	1	30	
4	4	2023-05-21	CUST004	Male	37	Clothing	1	500	
5	5	2023-05-06	CUST005	Male	30	Beauty	2	50	
6	6	2023-04-25	CUST006	Female	45	Beauty	1	30	
7	7	2023-03-13	CUST007	Male	46	Clothing	2	25	
8	8	2023-02-22	CUST008	Male	30	Electronics	4	25	
9	9	2023-12-13	CUST009	Male	63	Electronics	2	300	
10	10	2023-10-07	CUST010	Female	52	Clothing	4	50	
11	11	2023-02-14	CUST011	Male	23	Clothing	2	50	
12	12	2023-10-30	CUST012	Male	35	Beauty	3	25	



This screenshot shows the same Snowflake UI setup as the first one, but with a different SQL query in the phindy.sql worksheet:

```
1 | SELECT transaction_id,
2 |         date,
3 |         customer_id
4 |     FROM sales.retail.retail_sale_dataset;
5 | 
```

The Results section displays a table with 10 rows, showing the transaction ID, date, and customer ID for each record. The data corresponds to the first 10 entries in the previous screenshot.

#	TRANSACTION_ID	DATE	CUSTOMER_ID
1		2023-11-24	CUST001
2		2023-02-27	CUST002
3		2023-01-13	CUST003
4		2023-05-21	CUST004
5		2023-05-06	CUST005
6		2023-04-25	CUST006
7		2023-03-13	CUST007
8		2023-02-22	CUST008
9		2023-12-13	CUST009
10		2023-10-07	CUST010

## 2. SELECT DISTINCT Statement

The screenshot shows the AWS Glue Data Preview interface. In the top navigation bar, it says "Home" and "Load sample data from AWS S3 with SQL". The current file is "phindy.sql". The database is "ACCOUNTADMIN" and the compute type is "COMPUTE\_WH (X-Small)". The code in the editor is:

```
1 | SELECT DISTINCT product_category
2 | FROM sales.retail.retail_sale_dataset;
```

The results table shows the following data:

PRODUCT_CATEGORY
Clothing
Beauty
Electronics

There are 3 rows and the query took 22ms.

The screenshot shows the AWS Glue Data Preview interface. In the top navigation bar, it says "Home" and "Load sample data from AWS S3 with SQL". The current file is "phindy.sql". The database is "ACCOUNTADMIN" and the compute type is "COMPUTE\_WH (X-Small)". The code in the editor is:

```
1 | SELECT DISTINCT gender
2 | FROM sales.retail.retail_sale_dataset;
```

The results table shows the following data:

GENDER
Male
Female

There are 2 rows and the query took 78ms.

## 3. WHERE Clause

The screenshot shows the AWS Glue Data Preview interface. In the top navigation bar, it says "Home" and "Load sample data from AWS S3 with SQL". The current file is "phindy.sql". The database is "ACCOUNTADMIN" and the compute type is "COMPUTE\_WH (X-Small)". The code in the editor is:

```
1 | SELECT *
2 | FROM sales.retail.retail_sale_dataset
3 | WHERE age>40;
```

The results table shows the following data:

TRANSACTION_ID	DATE	CUSTOMER_ID	GEND	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT
3	2023-01-13	CUST003	Male	50	Electronics	1	30
6	2023-04-25	CUST006	Female	45	Beauty	1	30
7	2023-03-13	CUST007	Male	46	Clothing	2	25
9	2023-12-13	CUST009	Male	63	Electronics	2	300
10	2023-10-07	CUST010	Female	52	Clothing	4	50
14	2023-01-17	CUST014	Male	64	Clothing	4	30
15	2023-01-16	CUST015	Female	42	Electronics	4	500
18	2023-04-30	CUST018	Female	47	Electronics	2	25
19	2023-09-16	CUST019	Female	62	Clothing	2	25
21	2023-01-14	CUST021	Female	50	Beauty	1	500
24	2023-11-29	CUST024	Female	49	Clothing	1	300

There are 534 rows and the query took 71ms.

My W... ⋮

Load sample data from AWS S3 with SQL phindy.sql +

My Workspace > phindy.sql

ACCOUNTADMIN COMPUTE\_WH (X-Small) Choose database ...

```
1 | SELECT *
2 | FROM sales.retail.retail_sale_dataset
3 | WHERE price_per_unit BETWEEN 100 AND 500;
```

Results (just now)

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL
1	2	2023-02-27	CUST002	Female	26	Clothing	2	500	
2	4	2023-05-21	CUST004	Male	37	Clothing	1	500	
3	9	2023-12-13	CUST009	Male	63	Electronics	2	300	
4	13	2023-08-05	CUST013	Male	22	Electronics	3	500	
5	15	2023-01-16	CUST015	Female	42	Electronics	4	500	
6	16	2023-02-17	CUST016	Male	19	Clothing	3	500	
7	20	2023-11-05	CUST020	Male	22	Clothing	3	300	
8	21	2023-01-14	CUST021	Female	50	Beauty	1	500	
9	24	2023-11-29	CUST024	Female	49	Clothing	1	300	
10	26	2023-10-07	CUST026	Female	28	Electronics	2	500	
11	28	2023-04-23	CUST028	Female	43	Beauty	1	500	

My W... ⋮

Load sample data from AWS S3 with SQL phindy.sql +

My Workspace > phindy.sql

ACCOUNTADMIN COMPUTE\_WH (X-Small) Choose database ...

```
1 | SELECT *
2 | FROM sales.retail.retail_sale_dataset
3 | WHERE product_category = 'Beauty' OR product_category = 'Electronics';
```

Results (just now)

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	
2	3	2023-01-13	CUST003	Male	50	Electronics	1	30	
3	5	2023-05-06	CUST005	Male	30	Beauty	2	50	
4	6	2023-04-25	CUST006	Female	45	Beauty	1	30	
5	8	2023-02-22	CUST008	Male	30	Electronics	4	25	
6	9	2023-12-13	CUST009	Male	63	Electronics	2	300	
7	12	2023-10-30	CUST012	Male	35	Beauty	3	25	
8	13	2023-08-05	CUST013	Male	22	Electronics	3	500	
9	15	2023-01-16	CUST015	Female	42	Electronics	4	500	
10	18	2023-04-30	CUST018	Female	47	Electronics	2	25	
11	21	2023-01-14	CUST021	Female	50	Beauty	1	500	

My W... ⋮

Load sample data from AWS S3 with SQL phindy.sql +

My Workspace > phindy.sql

ACCOUNTADMIN COMPUTE\_WH (X-Small) Choose database ...

```
1 | SELECT *
2 | FROM sales.retail.retail_sale_dataset
3 | WHERE NOT product_category = 'Clothing' ;
```

Results (1 minute ago)

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	TOTAL
1	1	2023-11-24	CUST001	Male	34	Beauty	3	50	1
2	2	2023-02-27	CUST002	Female	26	Clothing	2	500	10
3	3	2023-01-13	CUST003	Male	50	Electronics	1	30	
4	4	2023-05-21	CUST004	Male	37	Clothing	1	500	5
5	5	2023-05-06	CUST005	Male	30	Beauty	2	50	1
6	6	2023-04-25	CUST006	Female	45	Beauty	1	30	
7	7	2023-03-13	CUST007	Male	46	Clothing	2	25	
8	8	2023-02-22	CUST008	Male	30	Electronics	4	25	1
9	9	2023-12-13	CUST009	Male	63	Electronics	2	300	6
10	10	2023-10-07	CUST010	Female	52	Clothing	4	50	2
11	11	2023-04-24	CUST011	Male	23	Clothing	2	50	1

The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with icons for file management, database exploration, and other workspace functions. The main area shows a SQL query in the editor:

```
1 | SELECT *  
2 | FROM sales.retail.retail_sale_dataset  
3 | WHERE quantity >= 3;
```

The results pane displays a table with 504 rows, showing transaction details like Transaction ID, Date, Customer ID, Gender, Age, Product Category, Quantity, and Price per Unit. The table has columns for each of these metrics.

## 4. Aggregate Functions

This screenshot shows the same AWS Glue Data Preview interface. A new query is run:

```
1 | SELECT COUNT(*) AS total_transactions  
2 | FROM sales.retail.retail_sale_dataset;  
3 |
```

The results show a single row with the value 1000 under the column labeled TOTAL\_TRANSACTIONS. A 'Download results' button is visible in the results pane.

This screenshot shows the AWS Glue Data Preview interface again. A query using the AVG function is run:

```
1 | SELECT AVG(age) AS Average_Age  
2 | FROM sales.retail.retail_sale_dataset;  
3 |
```

The results show a single row with the value 41.392000 under the column labeled AVERAGE\_AGE.

The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with icons for Home, My Workspace, Worksheets, and Database Explorer. The main area shows a SQL query in the editor:

```
1 | SELECT SUM(quantity) AS Total_Quantity
2 | FROM sales.retail.retail_sale_dataset;
3 |
```

The results pane displays a single row of data:

	TOTAL_QUANTITY
1	2514

At the bottom right of the results pane, it says "1 row 65ms".

The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with icons for Home, My Workspace, Worksheets, and Database Explorer. The main area shows a SQL query in the editor:

```
1 | SELECT MAX(total_amount) AS Max_Total_Amount
2 | FROM sales.retail.retail_sale_dataset;
3 |
4 |
```

The results pane displays a single row of data:

	MAX_TOTAL_AMOUNT
1	2000

At the bottom right of the results pane, it says "1 row 31ms".

The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with icons for Home, My Workspace, Worksheets, and Database Explorer. The main area shows a SQL query in the editor:

```
1 | SELECT MIN(price_per_unit) AS Min_Price_per_Unit
2 | FROM sales.retail.retail_sale_dataset;
3 |
```

The results pane displays a single row of data:

	MIN_PRICE_PER_UNIT
1	25

At the bottom right of the results pane, it says "1 row 29ms".

## 5. GROUP BY Statement

The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with 'My W...', 'Search for files', 'Add new', 'phindy.sql' (selected), and 'Untitled.sql'. Below that is a 'Worksheets' section with 'Search for files', 'Owned', 'Shared with you', and 'Scratchpad (Empty)'. The main area has tabs for 'Home', 'Load sample data from AWS S3 with SQL', and 'phindy.sql'. The 'phindy.sql' tab is active, displaying the following SQL code:

```
1 | SELECT product_category,
2 |         COUNT(*) AS Transaction_Count
3 |     FROM sales.retail.retail_sale_dataset
4 |     GROUP BY product_category;
5 | 
```

Below the code is a 'Results (1 minute ago)' section. It has 'Table' and 'Chart' tabs, with 'Table' selected. The table shows the following data:

PRODUCT_CATEGORY	TRANSACTION_COUNT
Clothing	351
Beauty	307
Electronics	342

At the bottom right of the results table are icons for 'Feedback', 'Copy', 'Print', and 'Download'.

This screenshot shows the same AWS Glue Data Preview interface. The 'phindy.sql' tab is active, displaying the following SQL code:

```
1 | SELECT product_category,
2 |         AVG(price_per_unit) AS Average_Price
3 |     FROM sales.retail.retail_sale_dataset
4 |     GROUP BY product_category;
5 | 
```

The 'Results (just now)' section shows the following data:

PRODUCT_CATEGORY	AVERAGE_PRICE
Beauty	184.055375
Clothing	174.287749
Electronics	181.900585

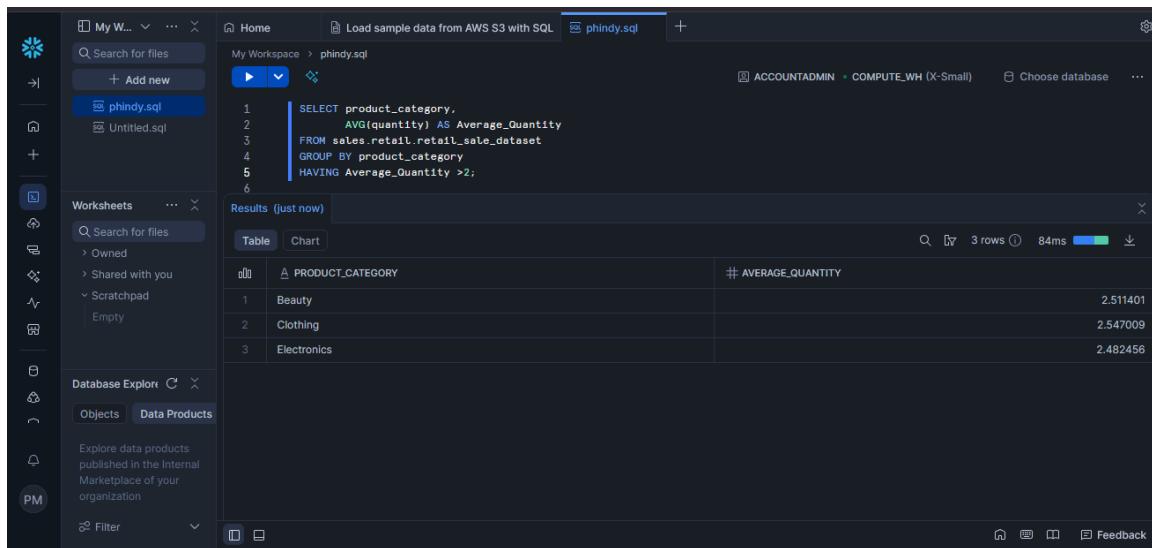
The screenshot shows the AWS Glue Data Preview interface again. The 'phindy.sql' tab is active, displaying the following SQL code:

```
1 | SELECT product_category,
2 |         SUM(total_amount) AS Total_Revenue
3 |     FROM sales.retail.retail_sale_dataset
4 |     GROUP BY product_category
5 |     HAVING Total_Revenue > 1000;
6 | 
```

The 'Results (just now)' section shows the following data:

PRODUCT_CATEGORY	TOTAL_REVENUE
Beauty	143515
Clothing	155580
Electronics	156905

## 6. HAVING Clause

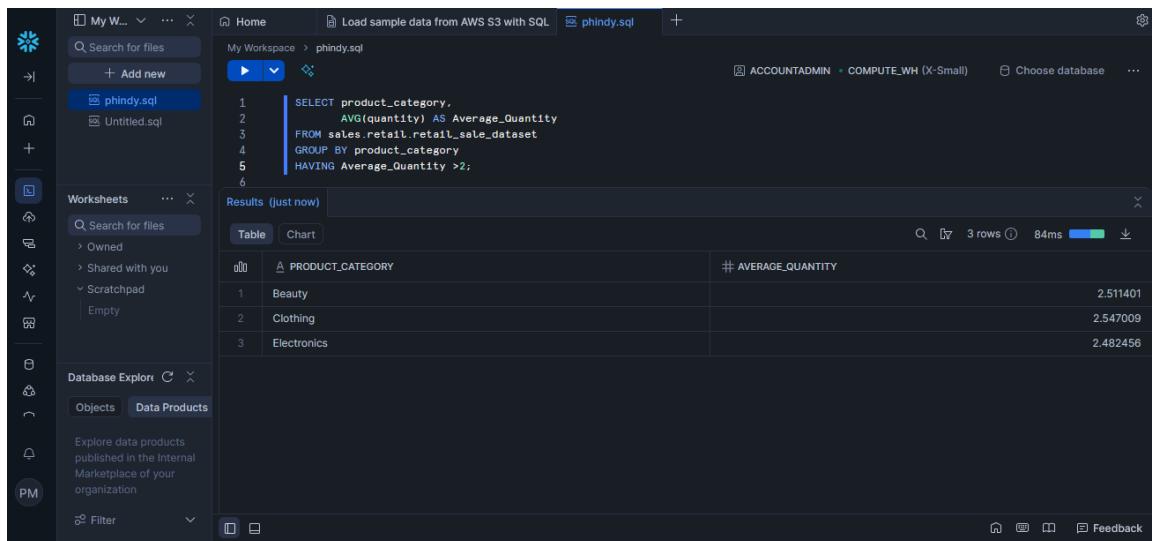


The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with 'Worksheets' and 'Database Explorer'. The main area shows a SQL worksheet named 'phindy.sql' with the following code:

```
1 | SELECT product_category,
2 |         AVG(quantity) AS Average_Quantity
3 |     FROM sales.retail.retail_sale_dataset
4 |     GROUP BY product_category
5 |     HAVING Average_Quantity >2;
```

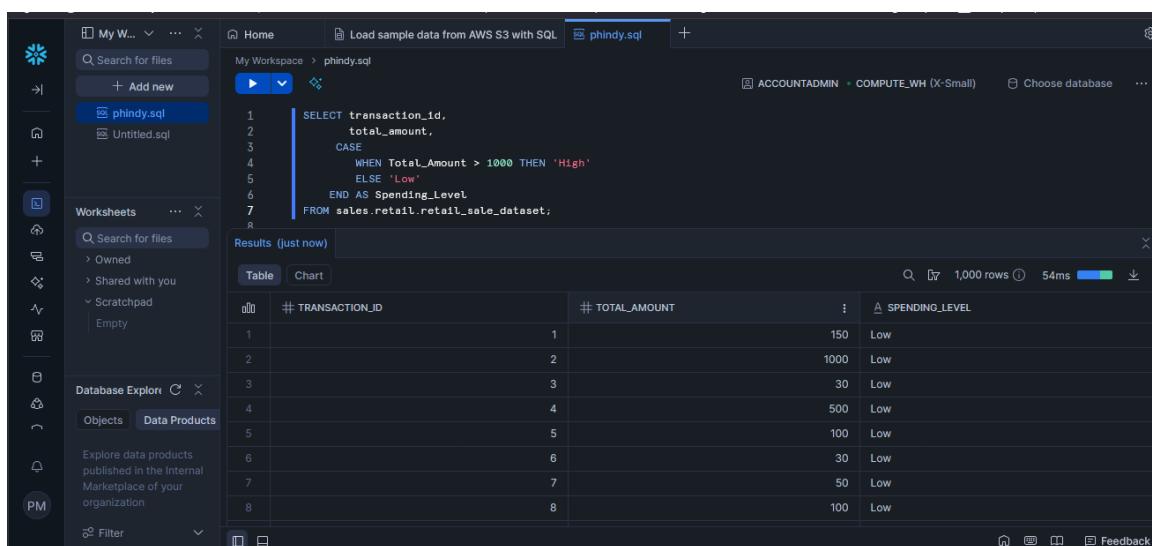
The results table shows three rows of data:

PRODUCT_CATEGORY	AVERAGE_QUANTITY
Beauty	2.511401
Clothing	2.547009
Electronics	2.482456



This screenshot is identical to the one above, showing the same SQL code and results table for the HAVING clause example.

## 7. CASE Statement



The screenshot shows the AWS Glue Data Preview interface. On the left, there's a sidebar with 'Worksheets' and 'Database Explorer'. The main area shows a SQL worksheet named 'phindy.sql' with the following code:

```
1 | SELECT transaction_id,
2 |         total_amount,
3 |         CASE
4 |             WHEN Total_Amount > 1000 THEN 'High'
5 |             ELSE 'Low'
6 |         END AS Spending_Level
7 |     FROM sales.retail.retail_sale_dataset;
```

The results table shows 1,000 rows of 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

The screenshot shows a data analysis workspace with the following components:

- Left Sidebar:** Includes a search bar, a "Add new" button, and sections for "Worksheets" (Owned, Shared with you) and "Database Explorer".
- Central Area:** A SQL editor window titled "phindy.sql" containing the following code:

```
1 | SELECT
2 |     Customer_ID,
3 |     Age,
4 |     CASE
5 |         WHEN Age < 30 THEN 'Youth'
6 |         WHEN Age BETWEEN 30 AND 59 THEN 'Adult'
7 |         ELSE 'Senior'
8 |     END AS Age_Group
9 | FROM sales_retail.retail_sale_dataset;
```
- Results Window:** Titled "Results (just now)" showing a table with 7 rows of 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