

Phindile Mnisi

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

1. SELECT Statement

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

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

The results are displayed in a table format, showing 12 rows of data. The table has the following columns: TRANSACTION_ID, DATE, CUSTOMER_ID, GENDER, AGE, PRODUCT_CATEGORY, QUANTITY, and PRICE_PER_UNIT.

	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

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

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

The results are displayed in a table format, showing 10 rows of data. The table has the following columns: TRANSACTION_ID, DATE, and CUSTOMER_ID.

	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

2. SELECT DISTINCT Statement

The screenshot shows a SQL IDE interface. The query editor contains the following SQL statement:

```
1 SELECT DISTINCT product_category
2 FROM sales.retail.retail_sale_dataset;
3
```

The results pane shows the output of the query:

PRODUCT_CATEGORY
Clothing
Beauty
Electronics

The results pane also indicates 3 rows and a execution time of 22ms.

The screenshot shows a SQL IDE interface. The query editor contains the following SQL statement:

```
1 SELECT DISTINCT gender
2 FROM sales.retail.retail_sale_dataset;
3
```

The results pane shows the output of the query:

GENDER
Male
Female

The results pane also indicates 2 rows and a execution time of 78ms.

3. WHERE Clause

The screenshot shows a SQL IDE interface. The query editor contains the following SQL statement:

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

The results pane shows the output of the query:

#	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UNIT	#
1	3	2023-01-13	CUST003	Male	50	Electronics	1	30	
2	6	2023-04-25	CUST006	Female	45	Beauty	1	30	
3	7	2023-03-13	CUST007	Male	46	Clothing	2	25	
4	9	2023-12-13	CUST009	Male	63	Electronics	2	300	
5	10	2023-10-07	CUST010	Female	52	Clothing	4	50	
6	14	2023-01-17	CUST014	Male	64	Clothing	4	30	
7	15	2023-01-16	CUST015	Female	42	Electronics	4	500	
8	18	2023-04-30	CUST018	Female	47	Electronics	2	25	
9	19	2023-09-16	CUST019	Female	62	Clothing	2	25	
10	21	2023-01-14	CUST021	Female	50	Beauty	1	500	
11	24	2023-11-29	CUST024	Female	49	Clothing	1	300	

The results pane also indicates 534 rows and a execution time of 71ms.

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

ACCOUNTADMIN

COMPUTE_WH (X-Small)

Choose database

1

2

3

SELECT *

FROM sales.retail.retail_sale_dataset

WHERE price_per_unit BETWEEN 100 AND 500;

Results (just now)

Table

Chart

396 rows

69ms

	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	
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	

Feedback

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

ACCOUNTADMIN

COMPUTE_WH (X-Small)

Choose database

1

2

3

SELECT *

FROM sales.retail.retail_sale_dataset

WHERE product_category = 'Beauty' OR product_category = 'Electronics';

Results (just now)

Table

Chart

649 rows

78ms

	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	
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	

Feedback

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

ACCOUNTADMIN

COMPUTE_WH (X-Small)

Choose database

1

2

3

SELECT *

FROM sales.retail.retail_sale_dataset

WHERE NOT product_category = 'Clothing' ;

Results (1 minute ago)

Table

Chart

1,000 rows

69ms

	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	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	2023-02-14	CUST011	Male	23	Clothing	2	50	1

Feedback

The screenshot shows a SQL IDE interface. On the left, there's a sidebar with a search bar, a file explorer showing 'phindy.sql' and 'Untitled.sql', and a 'Database Explorer' section. The main workspace displays a SQL query: `SELECT * FROM sales.retail_retail_sale_dataset WHERE quantity >= 3;`. Below the query, the results are shown in a table format. The table has 11 rows and 10 columns: TRANSACTION_ID, DATE, CUSTOMER_ID, GENDER, AGE, PRODUCT_CATEGORY, QUANTITY, and PRICE_PER_UNIT. The results show various transactions with quantities ranging from 3 to 500.

#	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	8	2023-02-22	CUST008	Male	30	Electronics	4	25
3	10	2023-10-07	CUST010	Female	52	Clothing	4	50
4	12	2023-10-30	CUST012	Male	35	Beauty	3	25
5	13	2023-08-05	CUST013	Male	22	Electronics	3	500
6	14	2023-01-17	CUST014	Male	64	Clothing	4	30
7	15	2023-01-16	CUST015	Female	42	Electronics	4	500
8	16	2023-02-17	CUST016	Male	19	Clothing	3	500
9	17	2023-04-22	CUST017	Female	27	Clothing	4	25
10	20	2023-11-05	CUST020	Male	22	Clothing	3	300
11	23	2023-04-12	CUST023	Female	35	Clothing	4	30

4. Aggregate Functions

The screenshot shows the same SQL IDE interface. The SQL query is now: `SELECT COUNT(*) AS total_transactions FROM sales.retail_retail_sale_dataset;`. The results show a single row with the value 1000 for the column TOTAL_TRANSACTIONS.

#	TOTAL_TRANSACTIONS
1	1000

The screenshot shows the same SQL IDE interface. The SQL query is now: `SELECT AVG(age) AS Average_Age FROM sales.retail_retail_sale_dataset;`. The results show a single row with the value 41.392000 for the column AVERAGE_AGE.

#	AVERAGE_AGE
1	41.392000

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

My Workspace > phindy.sql

ACCOUNTADMIN • COMPUTE_WH (X-Small)

Choose database

1

2

3

SELECT SUM(quantity) AS Total_Quantity

FROM sales.retail.retail_sale_dataset;

Results (just now)

Table

Chart

1 row

65ms

TOTAL_QUANTITY

1

2514

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

My Workspace > phindy.sql

ACCOUNTADMIN • COMPUTE_WH (X-Small)

Choose database

1

2

3

4

SELECT MAX(total_amount) AS Max_Total_Amount

FROM sales.retail.retail_sale_dataset;

Results (just now)

Table

Chart

1 row

31ms

MAX_TOTAL_AMOUNT

1

2000

My W...

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

Search for files

Owned

Shared with you

Scratchpad

Empty

Database Explorer

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

Home

Load sample data from AWS S3 with SQL

phindy.sql

My Workspace > phindy.sql

ACCOUNTADMIN • COMPUTE_WH (X-Small)

Choose database

1

2

3

SELECT MIN(price_per_unit) AS Min_Price_per_Unit

FROM sales.retail.retail_sale_dataset;

Results (just now)

Table

Chart

1 row

29ms

MIN_PRICE_PER_UNIT

1

25

5. GROUP BY Statement

The screenshot shows a SQL IDE interface with a workspace named 'phindy.sql'. The SQL query is as follows:

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

The results are displayed in a table format, showing 3 rows and 414ms execution time:

PRODUCT_CATEGORY	TRANSACTION_COUNT
Clothing	351
Beauty	307
Electronics	342

The screenshot shows the same SQL IDE interface with a new query:

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

The results are displayed in a table format, showing 3 rows and 78ms execution time:

PRODUCT_CATEGORY	AVERAGE_PRICE
Beauty	184.055375
Clothing	174.287749
Electronics	181.900585

The screenshot shows the same SQL IDE interface with a query that includes a HAVING clause:

```
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;
```

The results are displayed in a table format, showing 3 rows and 78ms execution time:

PRODUCT_CATEGORY	TOTAL_REVENUE
Beauty	143515
Clothing	155580
Electronics	156905

6. HAVING Clause

The screenshot shows a SQL IDE interface. The query editor contains the following SQL 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;  
6
```

The results pane shows the output of the query:

PRODUCT_CATEGORY	AVERAGE_QUANTITY
Beauty	2.511401
Clothing	2.547009
Electronics	2.482456

This is a duplicate of the previous screenshot, showing the same SQL query and results in the IDE.

7. CASE Statement

The screenshot shows a SQL IDE interface. The query editor contains the following SQL 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;  
8
```

The results pane shows the output of the query:

TRANSACTION_ID	TOTAL_AMOUNT	SPENDING_LEVEL
1	1	Low
2	2	Low
3	3	Low
4	4	Low
5	5	Low
6	6	Low
7	7	Low
8	8	Low

My W...

...

×

Search for files

+ Add new

phindy.sql

Untitled.sql

Worksheets

...

×

Search for files

> Owned

> Shared with you

> Scratchpad

Empty

Database Explorer

↻

×

Objects

Data Products

Explore data products published in the Internal Marketplace of your organization

Filter

▼

Home

Load sample data from AWS S3 with SQL

phindy.sql

+

ACCOUNTADMIN

>

COMPUTE_WH (X-Small)

>

Choose database

...

My Workspace

>

phindy.sql

▶

▼

✕

1

2

3

4

5

6

7

8

9

10

SELECT

Customer_ID,

Age,

CASE

WHEN Age < 30 THEN 'Youth'

WHEN Age BETWEEN 30 AND 59 THEN 'Adult'

ELSE 'Senior'

END AS Age_Group

FROM sales.retail.retail_sale_dataset;

Results (just now)

×

Table

Chart

🔍

📄

1,000 rows

⌚

80ms

📄

📄

△

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

🏠

🖨

📄

📧

Feedback