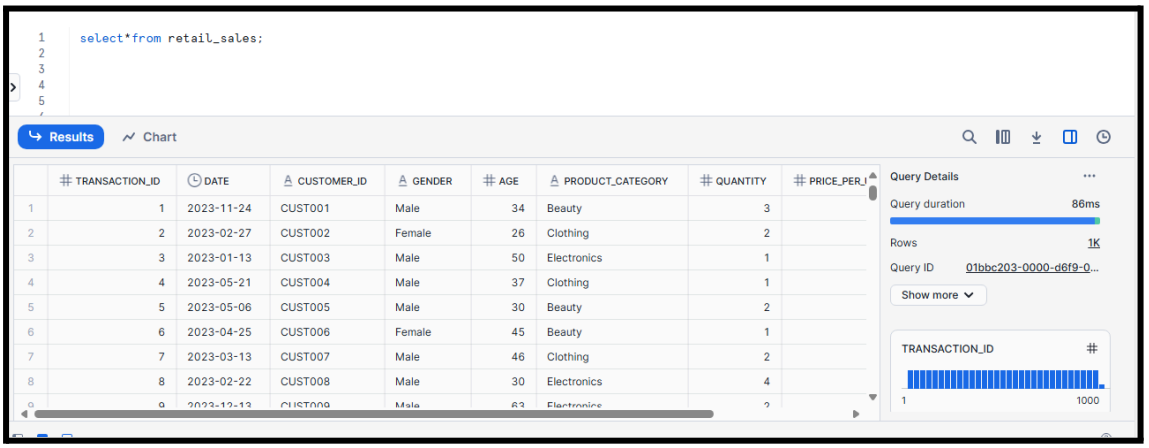
Practical 1: SQL Fundamentals (Snowflake-Basic SQL Syntax)

1. SELECT Statement

Q1. Display all columns for all transactions.

Expected output: All columns



```
select*from retail_sales;
```

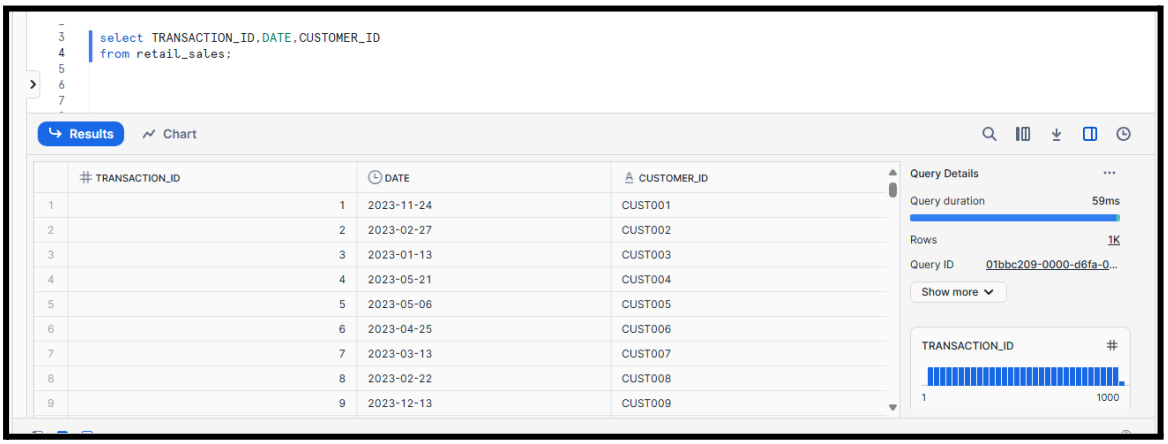
Results | Chart

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

Query Details
Query duration 86ms
Rows 1K
Query ID 01bbc203-0000-d6f9-0...
Show more
TRANSACTION_ID
1 1000

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 retail_sales;
```

Results | Chart

	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

Query Details
Query duration 59ms
Rows 1K
Query ID 01bbc209-0000-d6fa-0...
Show more
TRANSACTION_ID
1 1000

2. SELECT DISTINCT Statement

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

Expected output: Product Category

```
select distinct product_category
from retail_sales;
```

Results | Chart

	PRODUCT_CATEGORY
1	Beauty
2	Clothing
3	Electronics

Query Details
Query duration 66ms
Rows 3
Query ID 01bbc20c-0000-d6f9-0...
Show more

PRODUCT_CATEGORY
100% filled

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

Expected output: Gender

```
select distinct gender
from retail_sales;
```

Results | Chart

	GENDER
1	Male
2	Female

Query Details
Query duration 84ms
Rows 2
Query ID 01bbc212-0000-d701-0...
Show more

GENDER
100% filled

3. WHERE Clause

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

Expected output: All columns

```
select*from retail_sales
where AGE>40;
```

Results | Chart

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

Query Details
Query duration 57ms
Rows 534
Query ID 01bbc215-0000-d6f9-0...
Show more

TRANSACTION_ID
3 1000

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

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

Results | Chart

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

Query Details
Query duration 28ms
Rows 396
Query ID 01bbc218-0000-d6f9-0...
Show more
TRANSACTION_ID
2 994

Q7. Display all transactions where the Product Category is either 'Beauty' or 'Electronics'.
Expected output: All columns

```
select*from retail_sales
where product_category= 'Beauty' or product_category='Electronics';
```

Results | Chart

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

Query Details
Query duration 68ms
Rows 649
Query ID 01bbc21b-0000-d6f9-0...
Show more
TRANSACTION_ID
1 1000

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

```
select*from retail_sales
where not product_category='Clothing';
```

Results | Chart

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

Query Details
Query duration 90ms
Rows 649
Query ID 01bbc21d-0000-d6fa-0...
Show more
TRANSACTION_ID
1 1000

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

```
select*from retail_sales
where quantity>=3;
```

Results | Chart

	TRANSACTION_ID	DATE	CUSTOMER_ID	GENDER	AGE	PRODUCT_CATEGORY	QUANTITY	PRICE_PER_UI
1	1	2023-11-24	CUST001	Male	34	Beauty	3	
2	8	2023-02-22	CUST008	Male	30	Electronics	4	
3	10	2023-10-07	CUST010	Female	52	Clothing	4	
4	12	2023-10-30	CUST012	Male	35	Beauty	3	
5	13	2023-08-05	CUST013	Male	22	Electronics	3	
6	14	2023-01-17	CUST014	Male	64	Clothing	4	
7	15	2023-01-16	CUST015	Female	42	Electronics	4	
8	16	2023-02-17	CUST016	Male	19	Clothing	3	
9	17	2023-04-22	CUST017	Female	27	Clothing	4	

Query Details
Query duration 82ms
Rows 504
Query ID 01bbc220-0000-d701-0...
Show more
TRANSACTION_ID
1 1000

4. Aggregate Functions

Q10. Count the total number of transactions.
Expected output: Total_Transactions

```
select count(transaction_id) as Total_Transactions
from retail_sales;
```

Results | Chart

	TOTAL_TRANSACTIONS
1	1000

Query Details
Query duration 39ms
Rows 1
Query ID 01bbc223-0000-d6f8-0...
Show more
TOTAL_TRANSACTIONS
100% filled

Q11. Find the average Age of customers.
Expected output: Average_Age

```
select avg(age) as Average_Age
from retail_sales;
```

Results | Chart

	AVERAGE_AGE
1	41.392000

Query Details
Query duration 73ms
Rows 1
Query ID 01bbc224-0000-d6fa-0...
Show more
AVERAGE_AGE
100% filled

Q12. Find the total quantity of products sold.
Expected output: Total_Quantity

```
select sum(quantity) as Total_Quantity
from retail_sales;
```

Results | Chart

TOTAL_QUANTITY
2514

Query Details
Query duration 73ms
Rows 1
Query ID 01bbc226-0000-d6fa-0...
Show more

TOTAL_QUANTITY
100% filled

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 retail_sales;
```

Results | Chart

MAX_TOTAL_AMOUNT
2000

Query Details
Query duration 48ms
Rows 1
Query ID 01bbc228-0000-d6f9-0...
Show more

MAX_TOTAL_AMOUNT
100% filled

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 retail_sales;
```

Results | Chart

MIN_PRICE_PER_UNIT
25

Query Details
Query duration 71ms
Rows 1
Query ID 01bbc22a-0000-d6fa-0...
Show more

MIN_PRICE_PER_UNIT
100% filled

5. GROUP BY Statement

Q15. Find the number of transactions per Product Category.
Expected output: Product Category, Transaction_Count

```
select product_category, count(transaction_id) as transaction_count
from retail_sales
group by product_category;
```

Results | Chart

	PRODUCT_CATEGORY	TRANSACTION_COUNT
1	Beauty	307
2	Clothing	351
3	Electronics	342

Query Details
Query duration 115ms
Rows 3
Query ID 01bbc22c-0000-d700-0...
Show more

PRODUCT_CATEGORY
100% filled

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

```
select gender, sum(total_amount) as total_revenue
from retail_sales
group by gender;
```

Results | Chart

	GENDER	TOTAL_REVENUE
1	Male	223160
2	Female	232840

Query Details
Query duration 50ms
Rows 2
Query ID 01bbc22e-0000-d6f8-0...
Show more

GENDER
100% filled

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 retail_sales
group by product_category;
```

Results | Chart

	PRODUCT_CATEGORY	AVERAGE_PRICE
1	Beauty	184.055375
2	Clothing	174.287749
3	Electronics	181.900585

Query Details
Query duration 46ms
Rows 3
Query ID 01bbc230-0000-d6fa-0...
Show more

PRODUCT_CATEGORY
100% filled

6. HAVING Clause

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

Expected output: Product Category, Total_Revenue

```sql
select product_category, sum(total_amount) as total_revenue
from retail_sales
group by product_category
having sum(total_amount)>10000;
```

Results | Chart

	PRODUCT_CATEGORY	TOTAL_REVENUE
1	Beauty	143515
2	Clothing	155580
3	Electronics	156905

Query Details
Query duration 71ms
Rows 3
Query ID 01bbc232-0000-d701-0...
Show more

PRODUCT_CATEGORY
100% filled

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

Expected output: Product Category, Average_Quantity

```sql
select product_category, avg(quantity) as Average_Quantity
from retail_sales
group by product_category
having avg(quantity)>2;
```

Results | Chart

	PRODUCT_CATEGORY	AVERAGE_QUANTITY
1	Beauty	2.511401
2	Clothing	2.547009
3	Electronics	2.482456

Query Details
Query duration 69ms
Rows 3
Query ID 01bbc235-0000-d701-0...
Show more

PRODUCT_CATEGORY
100% filled

7. CASE Statement

Q20. Display a column called Spending_Level that shows 'High' if Total Amount > 1000, otherwise 'Low'.

Expected output: Transaction ID, Total Amount, Spending_Level

```sql
select transaction_id,total_amount,
case
when total_amount>1000 then 'High'
else 'Low'
end as spending_level
from retail_sales;
```

firebasestorage.googleapis.com

Results | Chart

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

Query Details
Query duration 31ms
Rows 1K
Query ID 01bbc23c-0000-d701-0...
Show more

TRANSACTION_ID

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

```sql
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 retail_sales;
```

Results | Chart

	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

Query Details

Query duration 28ms

Rows 1K

Query ID 01bbc243-0000-d6f9-0...

Show more

CUSTOMER_ID