

NOLWAZI NDIWENI

## PRACTICAL 1: BASIC SQL SYNTAX

### Question 1

▶

✓ 06:12 PM (14s)

1

```
-- Q1. Display all columns for all transactions.  
  
SELECT *  
FROM practical1.sales;
```



>  See performance (1)

Table ▼

+

	1 <sup>2</sup> <sub>3</sub> Transaction ID	 Date	A <sup>B</sup> <sub>C</sub> Customer ID	A <sup>B</sup> <sub>C</sub> Gender	1 <sup>2</sup> <sub>3</sub> Age	A <sup>B</sup> <sub>C</sub>
1	1	2023-11-24	CUST001	Male	34	B
2	2	2023-02-27	CUST002	Female	26	C
3	3	2023-01-13	CUST003	Male	50	E
4	4	2023-05-21	CUST004	Male	37	C
5	5	2023-05-06	CUST005	Male	30	B

### Question 2

FileEditViewRunHelpSQL ▾Tabs: ON ▾☆Last edit was now

▶ ▾

✓ 1 minute ago (1s)

1

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

SELECT `transaction id`,  
date,  
`customer id`  
FROM practical1.sales;



> [See performance \(1\)](#)

Table ▾

+

	<sup>1</sup> <sub>3</sub> transaction id	date	<sup>A</sup> <sub>C</sub> 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	

### QUESTION 3

File Edit View Run Help SQL ▾ Tabs: ON ▾ ☆ Last edit was 10 minutes...   Run all


 ▾ ✓ 07:50 PM (2s) 1


---


```
-- Q3. Display all the distinct product categories in the dataset.  
SELECT DISTINCT `product category`  
FROM practical1.sales;
```

>  See performance (1)

Table ▾ +

	 product category
1	Beauty
2	Clothing
3	Electronics

 ▾ 3 rows | 1.78s runtime

 This result is stored as `_sqldf` and can be used in other [Python](#) and [SQL](#) cells.

#### QUESTION 4

File Edit View Run Help SQL ▾ Tabs: ON ▾ ☆ Last edit was 1 minute ago

▶ ▾ ✓ 1 minute ago (2s) 1

```
-- Q4. Display all the distinct gender values in the dataset.  
SELECT DISTINCT gender  
FROM practical1.sales;
```

> [See performance \(1\)](#)

Table ▾ +

	<sup>A</sup> <sub>C</sub> gender
1	Male
2	Female

QUESTION 5

<div> <div>▶</div> <div>▼</div> <div>✓ 1 minute ago (2s)</div> <div>1</div> </div>		<pre>-- Q5. Display all transactions where the Age is greater than 40. SELECT * FROM practical1.sales WHERE age&gt;40;</pre>			
<div> <div>&gt;</div> <div> <div>📊</div> <div>See performance (1)</div> </div> </div>					
<div> <div>Table ▼</div> <div>+</div> </div>					
	📅 Date	A <sup>B</sup> <sub>C</sub> Customer ID	A <sup>B</sup> <sub>C</sub> Gender	1 <sup>2</sup> <sub>3</sub> Age	A <sup>B</sup> <sub>C</sub> Product Category
1	2023-01-13	CUST003	Male	50	Electronics
2	2023-04-25	CUST006	Female	45	Beauty
3	2023-03-13	CUST007	Male	46	Clothing
4	2023-12-13	CUST009	Male	63	Electronics
5	2023-10-07	CUST010	Female	52	Clothing
6	2023-01-17	CUST014	Male	64	Clothing

## QUESTION 6

▶

▼

✓

1 minute ago (2s)

1

```
-- Q6. Display all transactions where the Price per Unit is between 100 and
SELECT *
FROM practical1.sales
WHERE `price per unit` BETWEEN 100 AND 500;
```

> [See performance \(1\)](#)

Table ▼

+

	<sup>1</sup> <sub>3</sub> Transaction ID	Date	<sup>A</sup> <sub>C</sub> Customer ID	<sup>A</sup> <sub>C</sub> Gender	<sup>1</sup> <sub>3</sub> A
1	2	2023-02-27	CUST002	Female	
2	4	2023-05-21	CUST004	Male	
3	9	2023-12-13	CUST009	Male	
4	13	2023-08-05	CUST013	Male	
5	15	2023-01-16	CUST015	Female	
6	16	2023-02-17	CUST016	Male	
7	20	2023-11-05	CUST020	Male	

## QUESTION 7

✓ 1 minute ago (3s)

1

```
-- Q7. Display all transactions where the Product Category is either 'Beauty' or 'Electronics'
SELECT *
FROM practical1.sales
WHERE `product category` IN ('Beauty','Electronics');
```

> [See performance \(1\)](#)

Table ▾

+

	Transaction ID	Date	Customer ID	Gender	Age
1	1	2023-11-24	CUST001	Male	34
2	3	2023-01-13	CUST003	Male	50
3	5	2023-05-06	CUST005	Male	30
4	6	2023-04-25	CUST006	Female	45
5	8	2023-02-22	CUST008	Male	30
6	9	2023-12-13	CUST009	Male	63

itor/queries?contextId=sal-editor&o=998271988143642

## QUESTION 8

```
-- Q8. Display all transactions where the Product Category is not 'Clothing'.
SELECT *
FROM practical1.sales
WHERE `product category` NOT IN('Clothing');
```


>  See performance (1)

Table <span>▼</span>		+					
	1 <sup>2</sup> <sub>3</sub> Transaction ID	📅 Date	A <sup>B</sup> <sub>C</sub> Customer ID	A <sup>B</sup> <sub>C</sub> Gender	1 <sup>2</sup> <sub>3</sub> Age	A <sup>B</sup> <sub>C</sub> Product	
9	15	2023-01-16	CUST015	Female	42	Electronics	
10	18	2023-04-30	CUST018	Female	47	Electronics	
11	21	2023-01-14	CUST021	Female	50	Beauty	
12	25	2023-12-26	CUST025	Female	64	Beauty	
13	26	2023-10-07	CUST026	Female	28	Electronics	
14	27	2023-08-03	CUST027	Female	38	Beauty	



QUESTION 9

1 minute ago (30s)

1

SQL

-- Q9. Display all transactions where the Quantity is greater than or equal to 3.  
SELECT\*  
FROM practical1.sales  
WHERE quantity>=3;

> [See performance \(1\)](#)

Table

+

	Customer ID	Gender	Age	Product Category	Quantity	Price per
1	JST001	Male	34	Beauty	3	
2	JST008	Male	30	Electronics	4	
3	JST010	Female	52	Clothing	4	
4	JST012	Male	35	Beauty	3	
5	JST013	Male	22	Electronics	3	
6	JST014	Male	64	Clothing	4	
7	JST015	Female	42	Electronics	4	
8	JST016	Male	19	Clothing	3	

QUESTION 10

File Edit View Run Help SQL Tabs: ON Last edit was now Run all Connected

Just now (4s) 1 SQL

```
-- Q10. Count the total number of transactions.  
SELECT COUNT(`Transaction ID`)AS Total_transactions  
FROM practical1.sales;
```

> [See performance \(1\)](#)

Table +

	1.2 Total_transactions
1	1000

### QUESTION 11

```
-- Q11. Find the average Age of customers.  
SELECT AVG(Age)AS Average_age  
FROM practical1.sales;
```

> [See performance \(1\)](#)

Table +

	1.2 Average_age
1	41.392

### QUESTION 12

✓ 1 minute ago (4s)

1

-- Q12. Find the total quantity of products sold.  
SELECT SUM(Quantity)AS Total\_Quantity  
FROM practical1.sales;  
> [See performance \(1\)](#)

Table ▾

+

	1 <sup>2</sup> <sub>3</sub> Total_Quantity
1	2514

### QUESTION 13

✓ 1 minute ago (4s)

1

-- Q13. Find the maximum Total Amount spent in a single transaction.  
SELECT MAX(`Total Amount`)AS Max\_Total\_Amt  
FROM practical1.sales;  
> [See performance \(1\)](#)

Table ▾

+

	1 <sup>2</sup> <sub>3</sub> Max_Total_Amt
1	2000

### QUESTION 14

✓ 1 minute ago (4s)

1

-- Q14. Find the minimum Price per Unit in the dataset.  
SELECT MIN(`Price per Unit`)AS Min\_Price\_Per\_Unit  
FROM practical1.sales;  
> [See performance \(1\)](#)

Table ▾

+

	<sup>1 2 3</sup> Min_Price_Per_Unit	
1	25	

## QUESTION 15

✓ 2 minutes ago (4s)

1

-- Q15. Find the number of transactions per Product Category.  
SELECT `Product category`,COUNT(`product category`)AS Transaction\_Count  
FROM practical1.sales  
GROUP BY `Product category`;  
> [See performance \(1\)](#)

Table ▾

+

	<sup>A B C</sup> Product category	<sup>1 2 3</sup> Transaction_Count	
1	Beauty	307	
2	Clothing	351	
3	Electronics	342	

## QUESTION 16

▶

✓ 1 minute ago (4s)

1

SQL

```
-- Q16. Find the total revenue (Total Amount) per gender.
SELECT Gender,SUM(`Total Amount`)AS Total_Revenue
FROM practical1.sales
GROUP BY Gender;
```

> [See performance \(1\)](#)

Table 

+

	<div>A<sup>B</sup>C</div> Gender	<div>1<sup>2</sup>3</div> Total_Revenue
1	Male	223160
2	Female	232840

## QUESTION 17

▶

✓ Just now (4s)

1

S

```
FROM practical1.sales
GROUP BY Gender;
```

```
-- Q17. Find the average Price per Unit per product category.
SELECT `Product category`,AVG(`Price per Unit`) AS Average_price
FROM practical1.sales
GROUP BY `Product category`;
```

> [See performance \(1\)](#)

Table 

+

	<div>A<sup>B</sup>C</div> Product category	<div>1.2</div> Average_price
1	Beauty	184.05537459283389
2	Clothing	174.28774928774928
3	Electronics	181.90058479532163

## QUESTION 18

▶

✓ Just now (5s)

1

SQL

🗑️

-- Q18. Find the total revenue per product category where total revenue is greater than 10,000.  
SELECT `Product Category`,  
 SUM(Quantity\*`Price per Unit`)AS Total\_Revenue  
FROM practical1.sales  
GROUP BY `Product Category`  
HAVING Total\_Revenue>10000;


>  See performance (1)

Table ▼

+

🔍

	<div>⌵⌶</div> Product Category	<div>⌵⌶</div> Total_Revenue
1	Beauty	143515
2	Clothing	155580
3	Electronics	156905

## QUESTION 19

Just now (5s)

1

SQL

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

SELECT `Product Category`,

AVG(Quantity)AS Average\_Quantity

FROM practical1.sales

GROUP BY `Product Category`

HAVING Average\_Quantity>2;

> [See performance \(1\)](#)

Table

	Product Category	Average_Quantity
1	Beauty	2.511400651465798
2	Clothing	2.547008547008547
3	Electronics	2.482456140350877

## QUESTION 20

1 minute ago (5s)

1

-- Q20. Display a column called Spending\_Level that shows 'High' if Total Amount > 1000  
-- otherwise 'Low'.  
SELECT `Transaction ID` ,  
      `Total Amount`,  
      CASE  
          WHEN `Total Amount`>1000 THEN 'High'  
          ELSE 'Low'  
      END AS Spending\_Level  
FROM practical1.sales;

> [See performance \(1\)](#)

Table

+

	Transaction ID	Total Amount	Spending_Level	
1	1	150	Low	
2	2	1000	Low	
3	3	30	Low	

QUESTION 21



Just now (6s)

1

-- 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  
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 practical1.sales;

> See performance (1)

Table  +

	<div>A<sup>B</sup><sub>C</sub> Customer ID</div>	<div>1<sup>2</sup><sub>3</sub> Age</div>	<div>A<sup>B</sup><sub>C</sub> Age_Group</div>	
1	CUST001	34	Adult	
2	CUST002	26	Youth	