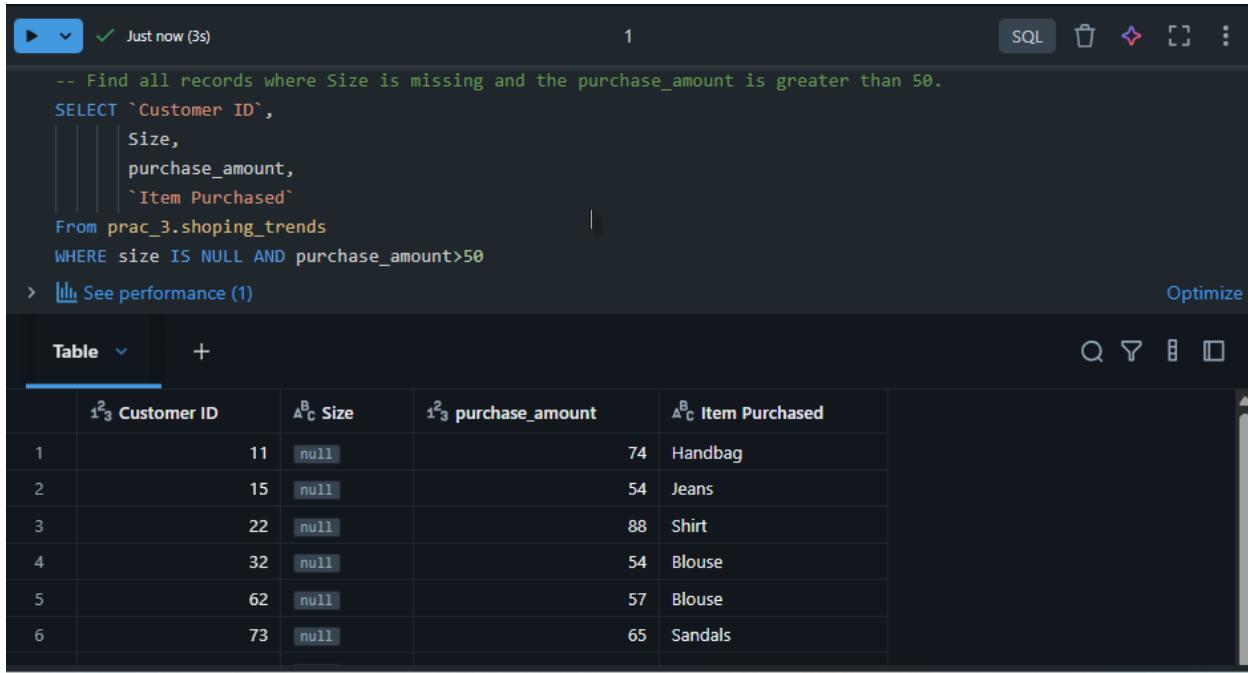


Nolwazi Ndiweni

## Practical 3

### Question 1



The screenshot shows a SQL query execution interface. The query is as follows:

```
-- Find all records where Size is missing and the purchase_amount is greater than 50.  
SELECT `Customer ID`,  
       Size,  
       purchase_amount,  
       `Item Purchased`  
FROM prac_3.shopping_trends  
WHERE size IS NULL AND purchase_amount > 50
```

The results table has the following data:

	Customer ID	Size	purchase_amount	Item Purchased
1	11	null	74	Handbag
2	15	null	54	Jeans
3	22	null	88	Shirt
4	32	null	54	Blouse
5	62	null	57	Blouse
6	73	null	65	Sandals

## Question 2

-- List the total number of purchases grouped by Season, treating NULL values as 'Unknown Season'.  
SELECT  
 IFNULL(`Season`, 'Unknown Season') AS Season,  
 COUNT(`Item Purchased`) AS Total\_number\_of\_purchases  
FROM `prac\_3.shoping\_trends`  
GROUP BY Season;

> See performance (1)      Optimize

	Season	Total_number_of_purchases
1	Winter	71
2	Spring	66
3	Unknown Seas...	26
4	Summer	58
5	Fall	50

## Question 3

-- Count how many customers used each Payment Method, treating NULLs as 'Not Provided'.  
SELECT  
 IFNULL(`Payment Method`, 'Not Provided') AS Payment\_Method,  
 COUNT(`Customer ID`) AS Customer\_Count  
FROM `prac\_3.shoping\_trends`  
GROUP BY Payment\_Method;

> See performance (1)      Optimize

	Payment_Method	Customer_C...
1	PayPal	51
2	Debit Card	42
3	Not Provided	30
4	Bank Transfer	38
5	Venmo	53
6	Credit Card	44
7	Cash	42

#### Question 4

A screenshot of a SQL editor interface. The top bar includes 'File', 'Edit', 'View', 'Run', 'Help', 'SQL', 'Tabs: ON', 'Connected', and 'Schedule'. A status bar at the bottom indicates 'Last edit was now' and '1 minute ago (1s)'. The main area contains the following SQL code:

```
SELECT
    `Customer ID`,
    `Promo Code Used`,
    `Review Rating`,
    `Item Purchased`
FROM shopping.trends_2
WHERE `Promo Code Used` IS NULL
AND `Review Rating` < 3.0;
```

The results pane shows a table with four columns: 'Customer ID', 'Promo Code Used', 'Review Rating', and 'Item Purchased'. Below the table, the message 'No rows returned' is displayed.

#### Question 6

A screenshot of a SQL editor interface. The top bar includes 'File', 'Edit', 'View', 'Run', 'Help', 'SQL', 'Tabs: ON', 'Connected', and 'Schedule'. A status bar at the bottom indicates 'Just now (5s)' and '1'. The main area contains the following SQL code:

```
-- 6. Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method.

SELECT
    `Location`,
    COUNT(*) AS `Total Purchases`
FROM shopping.trends_2
WHERE `Payment Method` IS NOT NULL
GROUP BY `Location`
HAVING COUNT(*) > 5;
```

The results pane shows a table with two columns: 'Location' and 'Total Purchases'. The data is as follows:

	Location	Total Purchases
1	Kentucky	79
2	Maine	77
3	Massachusetts	72

## Question 7

```
-- 7. Create a column Spender Category that classifies customers using CASE:  
-- 'High' if amount > 80, 'Medium' if BETWEEN 50 AND 80,  
-- 'Low' otherwise. Replace NULLs in purchase_amount with 0.  
  
SELECT  
    `Customer ID`,  
    COALESCE(`purchase amount (usd)`, 0) AS `purchase_amount`,  
    CASE  
        WHEN COALESCE(`purchase_amount`, 0) > 80 THEN 'High'  
        WHEN COALESCE(`purchase_amount`, 0) BETWEEN 50 AND 80 THEN 'Medium'  
        ELSE 'Low'  
    END AS `Spender Category`  
FROM shopping.trends_2;
```

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

Optimize

Table +



	Customer ID	purchase_amount	Spender Category
1	1	53	Medium
2	2	64	Medium

Q Y E F

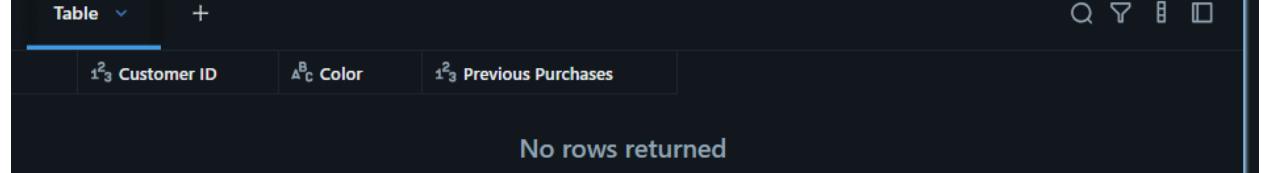
## Question 8

```
-- 8. Find customers who have no Previous Purchases value but whose Color is not NULL.  
  
SELECT  
    `Customer ID`,  
    Color,  
    `Previous Purchases`  
FROM shopping.trends_2  
WHERE `Previous Purchases` = 0  
    AND Color IS NOT NULL;
```

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

Optimize

Table +



	Customer ID	Color	Previous Purchases
No rows returned			

Q Y E F

## Question 9

-- 9. Group records by Frequency of Purchases and show the total amount spent per group, treating NULL frequencies as 'Unknown'.  
SELECT  
 COALESCE(`Frequency of Purchases`, 'Unknown') AS `Frequency of Purchases`,  
 SUM(COALESCE(`purchase amount (usd)`, 0)) AS `Total purchase\_amount`  
FROM shopping.trends\_2  
GROUP BY `Frequency of Purchases`;  
> See performance (1) Optimize

	Frequency of Purchases	Total purchase_amount
1	Fortnightly	32007
2	Weekly	31786
3	Annually	34419
4	Quarterly	33771
5	Bi-Weekly	33200
6	Monthly	32810

## Question 10

-- 10. Display a list of all Category values with the number of times each was purchased, excluding rows where Category is NULL.  
SELECT  
 Category,  
 COUNT(\*) AS `Total Purchases`  
FROM shopping.trends\_2  
WHERE Category IS NOT NULL  
GROUP BY Category;  
> See performance (1) Optimize

	Category	Total Purchases
1	Clothing	1737
2	Footwear	599
3	Outerwear	324
4	Accessories	1240

## Question 11

-- 11.Return the top5 Locations with the highest total purchase\_amount, replacing NULLs in amount with 0.

```
SELECT
    Location,
    SUM(COALESCE(`purchase amount (usd)`, 0)) AS `Total purchase_amount`
FROM shopping.trends_2
GROUP BY Location
ORDER BY `Total purchase_amount` DESC
LIMIT 5;
```

See performance (1)      Optimize

Table

	Location	Total purchase_amount
1	Montana	5784
2	Illinois	5617
3	California	5605
4	Idaho	5587
5	Nevada	5514

## Question 12

-- 12.Group customers by Gender and Size, and count how many entries have a NULL Color.

```
SELECT
    Gender,
    Size,
    COUNT(*) AS `Null Color Count`
FROM shopping.trends_2
WHERE Color IS NULL
GROUP BY Gender, Size;
```

See performance (1)      Optimize

Table

Gender	Size	Null Color Count
No rows returned		

### Question 13

Just now (1s) 1 SQL ⚙️ ⚡ ⚡

```
GROUP BY Gender, Size;

-- 13. Identify all Item Purchased where more than 3 purchases had NULL Shipping Type.
SELECT
    `Item Purchased`,
    COUNT(*) AS `NULL Shipping Type Count`
FROM shopping.trends_2
WHERE `Shipping Type` IS NULL
GROUP BY `Item Purchased`
HAVING COUNT(*) > 3;
| |
> See performance (1)
```

Generate (Ctrl + Optimize)

Table +

Item Purchased	NULL Shipping Type Count
Item A	5
Item B	3
Item C	2
Item D	1

### Question 14

1 minute ago (1s) 1 SQL ⚙️ ⚡ ⚡

```
-- 14. Show a count of how many customers per Payment Method have NULL Review Rating.
SELECT
    `Payment Method`,
    COUNT(*) AS `Missing Review Rating Count`
FROM shopping.trends_2
WHERE `Review Rating` IS NULL
GROUP BY `Payment Method`;
| |
> See performance (1)
```

Optimize

Table +

Payment Method	Missing Review Rating Count
Method A	100
Method B	50
Method C	20
Method D	10

## Question 15

The screenshot shows a SQL editor interface with the following details:

- Toolbar:** Includes icons for play, stop, refresh, and other database operations.
- Status Bar:** Shows "Just now (2s)" and a progress bar indicating the query took 1 second to run.
- Text Area:** Displays the SQL query:

```
-- 15. Group by Category and return the average Review Rating, replacing NULLs with 0, and filter only where
-- average is greater than 3.5.

SELECT
    Category,
    AVG(COALESCE(`Review Rating`, 0)) AS `Average Review Rating`
FROM shopping.trends_2
GROUP BY Category
HAVING AVG(COALESCE(`Review Rating`, 0)) > 3.5;
```
- Buttons:** "Generate (Ctrl + I)" and "Optimize".
- Table View:** A grid showing the results of the query. The columns are "Category" and "Average Review Rating". The rows are numbered 1 to 4, corresponding to the categories: Clothing, Footwear, Outerwear, and Accessories. The "Average Review Rating" column contains floating-point numbers.
- Table Options:** Buttons for "Table", "Generate (Ctrl + I)", "Optimize", and other table-related functions.

	Category	Average Review Rating
1	Clothing	3.7231433506044884
2	Footwear	3.7906510851419055
3	Outerwear	3.746913580246914
4	Accessories	3.7686290322580676