

BrightLight Data Analytics Coding Practical 3

--Q1. Find all records where Size is missing and the purchase_amount is greater than 50.

-- Expected Columns: Customer ID, Size, purchase_amount, Item Purchased

The screenshot shows a data analytics workspace titled 'PRACTICAL3.COOLESTDrip'. The code editor contains the following SQL query:

```
1 SELECT* FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS LIMIT 10;
2 --
3 --Q1. Find all records where Size is missing and the purchase_amount is greater than 50.
4 -- Expected Columns: Customer ID, Size, purchase_amount, Item Purchased
5
6 SELECT customer_id,
7       size,
8       purchase_amount,
9       item_purchased
10      FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
11     WHERE size IS NULL AND purchase_amount > 50;
12
13 --
14 -- Q2 List the total number of purchases grouped by Season, treating NULL values as 'Unknown Season'.
15 -- Expected Columns: Season, Total Purchases
16
17 SELECT COUNT(*) AS Total_Purchases
```

The results table displays the following data:

#	CUSTOMER_ID	SIZE	PURCHASE_AMOUNT	ITEM_PURCHASED
1	11	null	74.0	Handbag
2	15	null	54.0	Jeans
3	22	null	88.0	Shirt
4	32	null	54.0	Blouse
5	62	null	57.0	Blouse

-- Q2 List the total number of purchases grouped by Season, treating NULL values as 'Unknown Season'.

-- Expected Columns: Season, Total Purchases

The screenshot shows a database query editor with the following code:

```
14 -- Q2 List the total number of purchases grouped by Season, treating NULL values
15 -- as 'Unknown Season'.
16 -- Expected Columns: Season, Total Purchases
17
18 | SELECT COUNT(*) AS Total_Purchases,
19 |     IFNULL(SEASON, 'Unknown') AS Season
20 | FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
21 | GROUP BY SEASON;
22 |
23 | -- Each Payment Method, treating NULLs as 'Not Provided'.
24 | -- Expected Columns: Payment Method, Customer Count
25
26 | SELECT COUNT(customer_id) AS Customer_Count,
27 |     IFNULL(PAYMENT_METHOD, 'Not Provided') AS Payment_Method
28 | FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
29 | GROUP BY payment_method;
```

The results table has two columns: '# TOTAL_PURCHASES' and 'SEASON'. The data is:

# TOTAL_PURCHASES	SEASON
65	Summer
80	Winter
55	Fall
27	Unknown
73	Spring

-- Q3. Count how many customers used each Payment Method, treating NULLs as 'Not Provided'.

-- Expected Columns: Payment Method, Customer Count

The screenshot shows a database query editor with the following code:

```
21
22 | -- Q3. Count how many customers used each Payment Method, treating NULLs as 'Not Provided'.
23 | -- Expected Columns: Payment Method, Customer Count
24
25 | SELECT COUNT(customer_id) AS Customer_Count,
26 |     IFNULL(PAYMENT_METHOD, 'Not Provided') AS Payment_Method
27 | FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
28 | GROUP BY payment_method;
29
30
31 | --Q4 Show customers where Promo Code Used is NULL and Review Rating is below 3.0.
32 | -- Expected Columns: Customer ID, Promo Code Used, Review Rating, Item Purchased
33
34 | SELECT customer_id,
35 |     ITEM_PURCHASED,
36 |     PROMO_CODE_USED
37 | FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
```

The results table has two columns: '# TOTAL_PURCHASES' and 'SEASON'. The data is:

# TOTAL_PURCHASES	SEASON
65	Summer
80	Winter
55	Fall
27	Unknown
73	Spring

--Q4 Show customers where Promo Code Used is NULL and Review Rating is below 3.0.

-- Expected Columns: Customer ID, Promo Code Used, Review Rating, Item Purchased

The screenshot shows a Jupyter Notebook cell with the following code:

```
2025-10-15 9:38pm Practical1 Practical2 Practical3 + ▾
ACCOUNTADMIN COMPUTE_WH (Medium) Share ▶
PRACTICAL3.COOLESTDrip Settings ▾
FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
GROUP BY payment_method;

--Q4 Show customers where Promo Code Used is NULL and Review Rating is below 3.0.
-- Expected Columns: Customer ID, Promo Code Used, Review Rating, Item Purchased

SELECT customer_id,
       ITEM_PURCHASED,
       PROMO_CODE_USED
  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
 WHERE PROMO_CODE_USED IS NULL AND REVIEW_RATING < 3.0;

-- Group customers by Shipping Type, and return the average purchase_amount, treating missing values as 0.
-- Expected Columns: Shipping Type, Average purchase_amount
```

The results table has columns `# TOTAL_PURCHASES` and `SEASON`, with the following data:

	# TOTAL_PURCHASES	SEASON
1	85	Summer
2	80	Winter
3	55	Fall
4	27	Unknown
5	73	Spring

-- Q5 Group customers by Shipping Type, and return the average purchase_amount, treating missing values as 0.

-- Expected Columns: Shipping Type, Average purchase_amount

The screenshot shows a Jupyter Notebook cell with the following code:

```
2025-10-15 9:38pm Practical1 Practical2 Practical3 + ▾
ACCOUNTADMIN COMPUTE_WH (Medium) Share ▶
PRACTICAL3.COOLESTDrip Settings ▾
       ITEM_PURCHASED,
       PROMO_CODE_USED
  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
 WHERE PROMO_CODE_USED IS NULL AND REVIEW_RATING < 3.0;

-- Q5 Group customers by Shipping Type, and return the average purchase_amount, treating missing values as 0.
-- Expected Columns: Shipping Type, Average purchase_amount

SELECT shipping_type, AVG(purchase_amount) AS Average_purchase_Amount
  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
 GROUP BY shipping_type;

-- Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method.
-- Expected Columns: Location, Total Purchases
```

The results table has columns `# SHIPPING_TYPE` and `AVERAGE_PURCHASE_AMOUNT`, with the following data:

	# SHIPPING_TYPE	AVERAGE_PURCHASE_AMOUNT
1	Standard	55.0000000
2	Express	60.3076923
3	Store Pickup	60.7317073
4	null	61.8695652
5	Free Shipping	60.2571429

--Q6 Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method.

-- Expected Columns: Location, Total Purchases

The screenshot shows a code editor with a dark theme. The code is written in SQL and includes comments explaining the purpose of each section. The results tab is selected, displaying a table with five rows. The columns are labeled 'LOCATION' and '# TOTAL_PURCHASES'. The data shows purchase counts for Maine (41), Kentucky (30), null (24), New York (31), and Oregon (30).

```
42 EXPECTED_COLUMNS: SHIPPING_TYPE, AVG(purchase_amount) AS Average_purchase_Amount
43
44 SELECT shipping_type, AVG(purchase_amount) AS Average_purchase_Amount
45 FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
46 GROUP BY shipping_type;
47
48 --Q6 Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method.
49 -- Expected Columns: Location, Total Purchases
50
51
52 SELECT location, COUNT(*) AS Total_Purchases
53 FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS
54 WHERE PAYMENT_METHOD IS NOT NULL
55 GROUP BY location
56 HAVING Total_Purchases >5;
57
```

LOCATION	# TOTAL_PURCHASES
Maine	41
Kentucky	30
null	24
New York	31
Oregon	30

-- Q7 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.

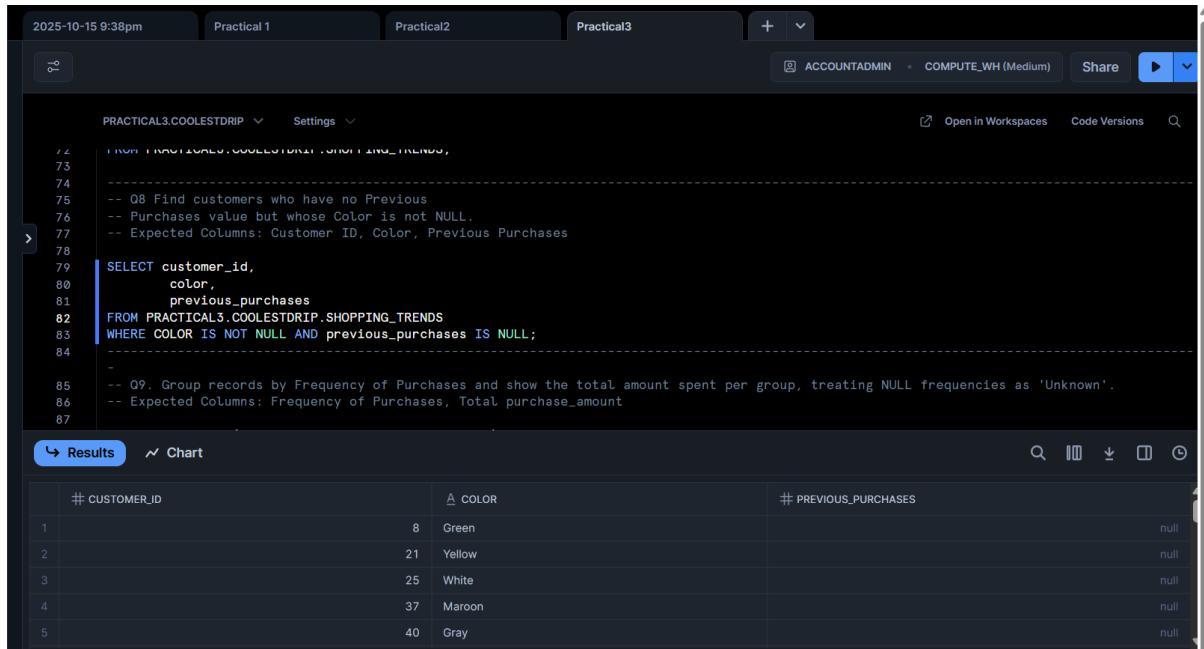
-- Expected Columns: Customer ID, purchase_amount, Spender Category

The screenshot shows a code editor with a dark theme. The code uses a CASE statement to classify customer purchase amounts into 'High', 'Medium', or 'Low'. It also handles NULL values by replacing them with 0. The results tab is selected, displaying a table with five rows. The columns are labeled 'CUSTOMER_ID', 'PURCHASE_AMOUNT', 'SPENDER_CATEGORY', and 'PURCHASE_AMOUNT'. The 'PURCHASE_AMOUNT' column contains the original values (20.0, 21.0, 27.0, 45.0, 80.0), while the 'SPENDER_CATEGORY' column shows the classification ('Low', 'Low', 'Low', 'Low', 'Medium').

```
58 -- Q7 Create a column Spender Category that classifies customers using CASE:
59 -- 'High' if amount > 80, 'Medium' if BETWEEN 50 AND 80,
60 -- 'Low' otherwise. Replace NULLs in purchase_amount with 0.
61 -- Expected Columns: Customer ID, purchase_amount, Spender Category
62
63
64 SELECT customer_id,
65     purchase_amount,
66     CASE
67         WHEN purchase_amount > 80 THEN 'High'
68         WHEN purchase_amount BETWEEN 50 AND 80 THEN 'Medium'
69         ELSE 'Low'
70     END AS Spender_Category,
71     IFNULL(purchase_amount,0) AS purchase_amount
72     FROM PRACTICAL3.COOLESTDRIPT.SHOPPING_TRENDS;
73
```

CUSTOMER_ID	PURCHASE_AMOUNT	SPENDER_CATEGORY	PURCHASE_AMOUNT
1	20.0	Low	20.0
2	21.0	Low	21.0
3	27.0	Low	27.0
4	45.0	Low	45.0
5	80.0	Medium	80.0

-
- Q8 Find customers who have no Previous Purchases value but whose Color is not NULL.
-- Expected Columns: Customer ID, Color, Previous Purchases



```

2025-10-15 9:38pm Practical 1 Practical2 Practical3 + ▾
ACCOUNTADMIN COMPUTE_WH (Medium) Share ▶

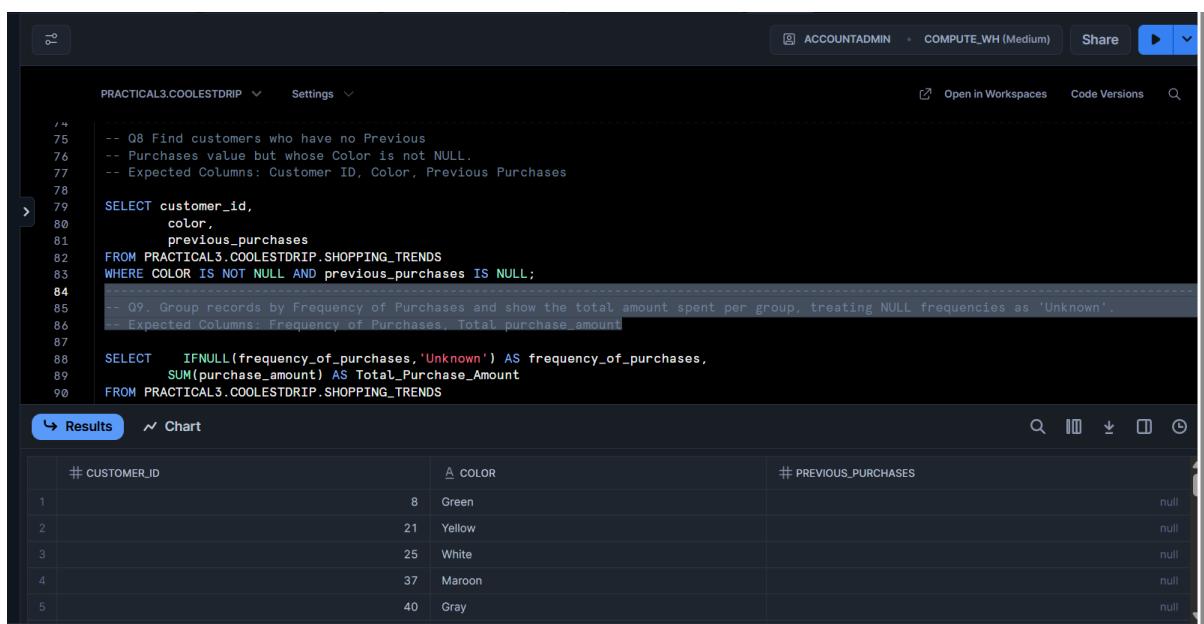
PRACTICAL3.COOLESTDrip Settings ▾
FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS;
-----+
75  -- Q8 Find customers who have no Previous
76  -- Purchases value but whose Color is not NULL.
77  -- Expected Columns: Customer ID, Color, Previous Purchases
78
79  SELECT customer_id,
80      color,
81      previous_purchases
82  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
83  WHERE COLOR IS NOT NULL AND previous_purchases IS NULL;
84
85  --
86  -- Q9. Group records by Frequency of Purchases and show the total amount spent per group, treating NULL frequencies as 'Unknown'.
87  -- Expected Columns: Frequency of Purchases, Total purchase_amount
88
89
90

```

Results

# CUSTOMER_ID	A COLOR	# PREVIOUS_PURCHASES
1	8 Green	null
2	21 Yellow	null
3	25 White	null
4	37 Maroon	null
5	40 Gray	null

-
- Q9. Group records by Frequency of Purchases and show the total amount spent per group, treating NULL frequencies as 'Unknown'.
-- Expected Columns: Frequency of Purchases, Total purchase_amount



```

2025-10-15 9:38pm Practical 1 Practical2 Practical3 + ▾
ACCOUNTADMIN COMPUTE_WH (Medium) Share ▶

PRACTICAL3.COOLESTDrip Settings ▾
-----+
75  -- Q8 Find customers who have no Previous
76  -- Purchases value but whose Color is not NULL.
77  -- Expected Columns: Customer ID, Color, Previous Purchases
78
79  SELECT customer_id,
80      color,
81      previous_purchases
82  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
83  WHERE COLOR IS NOT NULL AND previous_purchases IS NULL;
84
85  --
86  -- Q9. Group records by Frequency of Purchases and show the total amount spent per group, treating NULL frequencies as 'Unknown'.
87  -- Expected Columns: Frequency of Purchases, Total purchase_amount
88
89  SELECT IFNULL(frequency_of_purchases, 'Unknown') AS frequency_of_purchases,
90         SUM(purchase_amount) AS Total_Purchase_Amount
91  FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
92
93
94

```

Results

# CUSTOMER_ID	A COLOR	# PREVIOUS_PURCHASES
1	8 Green	null
2	21 Yellow	null
3	25 White	null
4	37 Maroon	null
5	40 Gray	null

-- Q10.Display a list of all Category values with the number of times each was purchased, excluding rows where Categoryis NULL.

-- Expected Columns: Category, Total Purchases

The screenshot shows a database interface with the following details:

- Database:** PRACTICAL3.COOLESTDrip
- Table:** SHOPPING_TRENDS
- Code:**

```
93 -- Q10.Display a list of all Category values with the number of times each was purchased, excluding rows where Categoryis NULL.
94 -- Expected Columns: Category, Total Purchases
95
96 SELECT category, SUM(PURCHASE_AMOUNT) AS Total_Purchases,
97   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
98 WHERE category IS NOT NULL
99 GROUP BY category;
100
101 -- Q11.Return the top 5 Locations with the highest total purchase_amount, replacing NULLs in amount with 0.
102 -- Expected Columns: Location, Total purchase_amount
103
104 SELECT location, IFNULL(SUM(PURCHASE_AMOUNT),0) AS Total_Purchase_Amount
105   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
106 GROUP BY location
107 ORDER BY IFNULL(SUM(PURCHASE_AMOUNT),0) DESC
108
109
110
```
- Results:** A table showing the total purchases for each category.

Category	Total Purchases
Outerwear	2880.0
Footwear	3733.0
Clothing	3022.0
Accessories	4242.0

-- Q11.Return the top 5 Locations with the highest total purchase_amount, replacing NULLs in amount with 0.

-- Expected Columns: Location, Total purchase_amount

The screenshot shows a database interface with the following details:

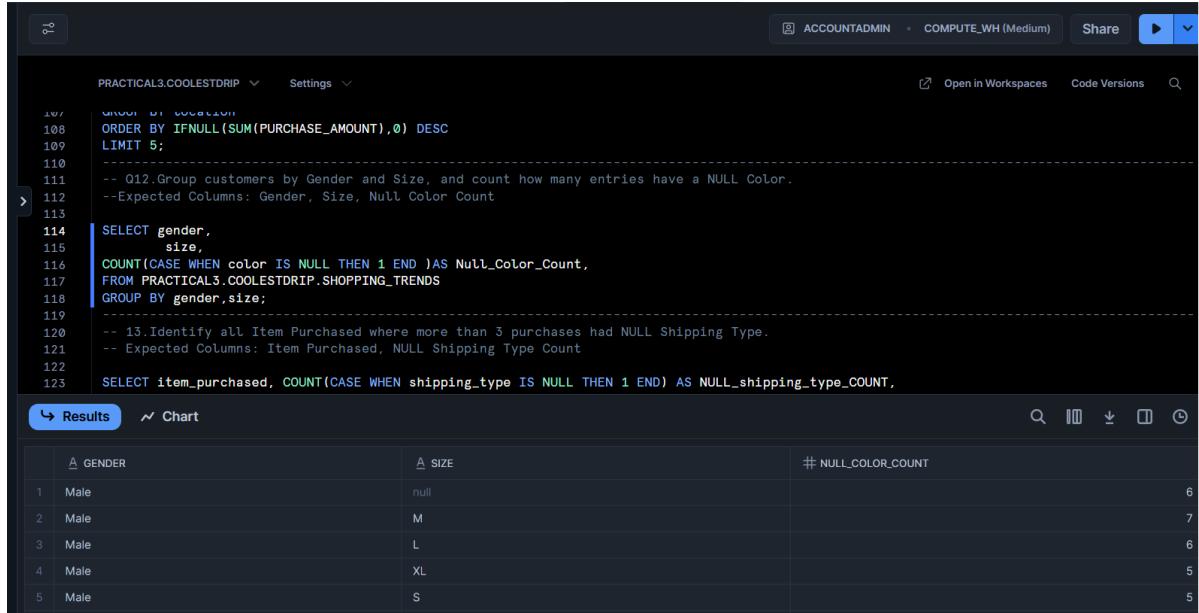
- Database:** PRACTICAL3.COOLESTDrip
- Table:** SHOPPING_TRENDS
- Code:**

```
93 -- Q10.Display a list of all Category values with the number of times each was purchased, excluding rows where Categoryis NULL.
94 -- Expected Columns: Category, Total Purchases
95
96 SELECT category, SUM(PURCHASE_AMOUNT) AS Total_Purchases,
97   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
98 WHERE category IS NOT NULL
99 GROUP BY category;
100
101 -- Q11.Return the top 5 Locations with the highest total purchase_amount, replacing NULLs in amount with 0.
102 -- Expected Columns: Location, Total purchase_amount
103
104 SELECT location, IFNULL(SUM(PURCHASE_AMOUNT),0) AS Total_Purchase_Amount
105   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
106 GROUP BY location
107 ORDER BY IFNULL(SUM(PURCHASE_AMOUNT),0) DESC
108
109 LIMIT 5;
110
```
- Results:** A table showing the top 5 locations with their total purchase amount.

Location	Total Purchase Amount
Maine	2294.0
Florida	1980.0
Massachusetts	1899.0
Rhode Island	1876.0
Kentucky	1798.0

-- Q12.Group customers by Gender and Size, and count how many entries have a NULL Color.

--Expected Columns: Gender, Size, Null Color Count



PRACTICAL3.COOLESTDrip Settings Open in Workspaces Code Versions Share

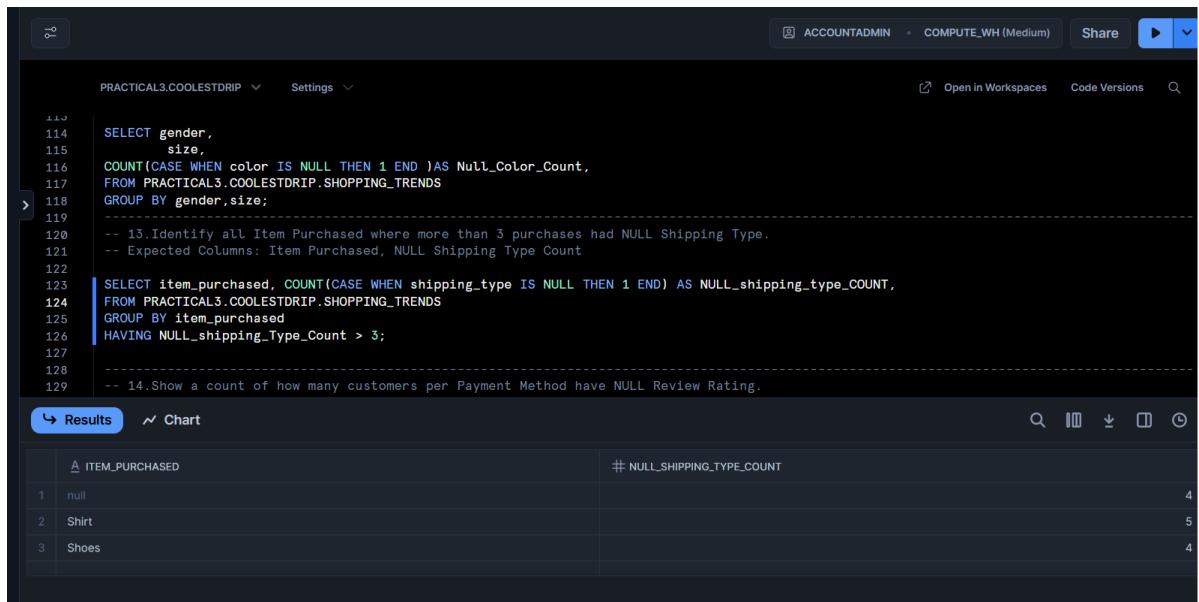
```
107 GROUP BY location
108 ORDER BY IFNULL(SUM(PURCHASE_AMOUNT),0) DESC
109 LIMIT 5;
110
111 -- Q12.Group customers by Gender and Size, and count how many entries have a NULL Color.
112 --Expected Columns: Gender, Size, Null Color Count
113
114 SELECT gender,
115      size,
116      COUNT(CASE WHEN color IS NULL THEN 1 END )AS Null_Color_Count,
117      FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
118      GROUP BY gender,size;
119
120 -- 13.Identify all Item Purchased where more than 3 purchases had NULL Shipping Type.
121 -- Expected Columns: Item Purchased, NULL Shipping Type Count
122
123 SELECT item_purchased, COUNT(CASE WHEN shipping_type IS NULL THEN 1 END ) AS NULL_shipping_type_COUNT,
```

↳ Results ▾ Chart

GENDER	SIZE	NULL_COLOR_COUNT
Male	null	6
Male	M	7
Male	L	6
Male	XL	5
Male	S	5

-- Q13.Identify all Item Purchased where more than 3 purchases had NULL Shipping Type.

-- Expected Columns: Item Purchased, NULL Shipping Type Count



PRACTICAL3.COOLESTDrip Settings Open in Workspaces Code Versions Share

```
114 SELECT gender,
115      size,
116      COUNT(CASE WHEN color IS NULL THEN 1 END )AS Null_Color_Count,
117      FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
118      GROUP BY gender,size;
119
120 -- 13.Identify all Item Purchased where more than 3 purchases had NULL Shipping Type.
121 -- Expected Columns: Item Purchased, NULL Shipping Type Count
122
123 SELECT item_purchased, COUNT(CASE WHEN shipping_type IS NULL THEN 1 END ) AS NULL_shipping_type_COUNT,
124      FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
125      GROUP BY item_purchased
126      HAVING NULL_shipping_Type_Count > 3;
127
128 -- 14.Show a count of how many customers per Payment Method have NULL Review Rating.
129
```

↳ Results ▾ Chart

ITEM_PURCHASED	NULL_SHIPPING_TYPE_COUNT
null	4
Shirt	5
Shoes	4

-- Q14.Show a count of how many customers per Payment Method have NULL Review Rating.

-- Expected Columns: Payment Method, Missing Review Rating Count

The screenshot shows a code editor with a dark theme. The code is written in SQL and includes comments for Q13 and Q14. The results tab is selected, displaying a table with two columns: PAYMENT_METHOD and #NULL REVIEW_RATING_COUNT. The data shows the count of customers for each payment method who have a null review rating.

PAYMENT_METHOD	#NULL REVIEW_RATING_COUNT
Credit Card	8
PayPal	3
Debit Card	7
null	2
Cash	4

-- Q15.Group by Category and return the average Review Rating, replacing NULLs with 0, and filter only where average is greater than 3.5.

-- Expected Columns: Category, Average Review Rating

The screenshot shows a code editor with a dark theme. The code is written in SQL and includes comments for Q15 and Q16. The results tab is selected, displaying a table with two columns: CATEGORY and #AVERAGE REVIEW RATING. The data shows the average review rating for each category, filtering only those with an average rating greater than 3.5.

CATEGORY	#AVERAGE REVIEW RATING
Outerwear	3.8173077
Footwear	3.6573770
null	3.7258065
Accessories	3.7338235

--Q16. List all Colors that are missing (NULL) in at least 2 rows and the average Age of customers for those rows.

-- Expected Columns: Color, Average Age



PRACTICAL3.COOLESTDrip ▾ Settings ▾ Open in Workspaces Code Versions

```
139 SELECT category, IFNULL(avg(review_rating),0) AS Average_Review_Rating,
140   FROM practical3.coolestdrip.shopping_trends
141   GROUP BY category
142   HAVING Average_Review_Rating > 3.5;
143
144 --Q16. List all Colors that are missing (NULL) in at least 2 rows and the average Age of customers for those rows.
145 -- Expected Columns: Color, Average Age
146
147 SELECT Color, avg(age) AS Average_Age
148   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
149   WHERE color IS NULL
150   GROUP BY color
151   HAVING COUNT(*) >= 2;
152
153 -- 17. Use CASE to create a column Delivery Speed: 'Fast' if Shipping Type is 'Express' or
154 -- 'Next Day Air', 'Slow' if 'Standard','Other' for all else including NULL. Then count how many customers fall into each category.
```

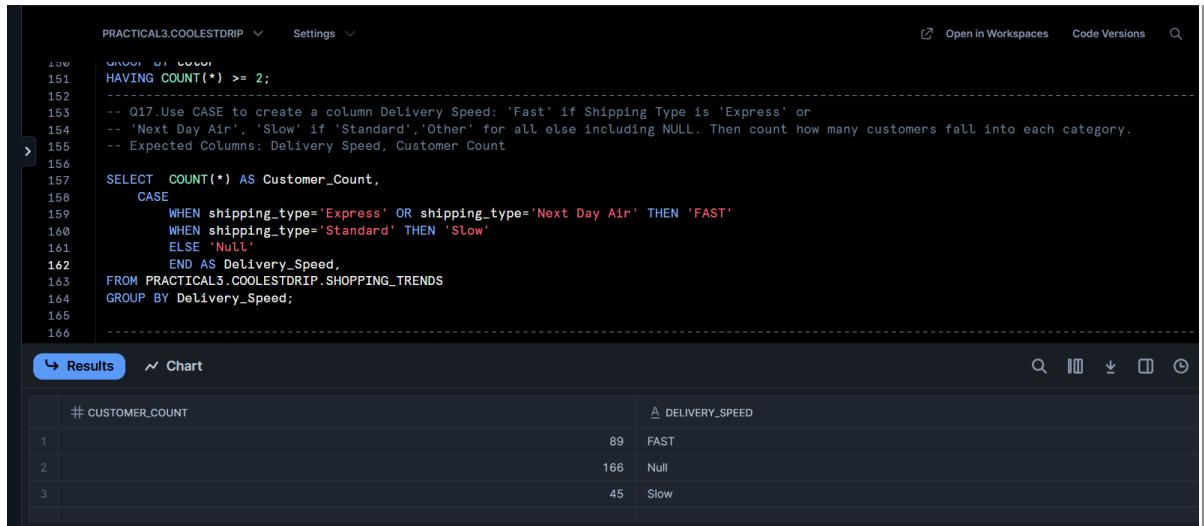
↳ Results ▾ Chart

A COLOR	# AVERAGE_AGE
1 null	47.8461538

--Q17. Use CASE to create a column Delivery Speed: 'Fast' if Shipping Type is 'Express' or

-- 'Next Day Air', 'Slow' if 'Standard','Other' for all else including NULL. Then count how many customers fall into each category.

-- Expected Columns: Delivery Speed, Customer Count



PRACTICAL3.COOLESTDrip ▾ Settings ▾ Open in Workspaces Code Versions

```
139 GROUP BY color
140 HAVING COUNT(*) >= 2;
141
142 -- Q17. Use CASE to create a column Delivery Speed: 'Fast' if Shipping Type is 'Express' or
143 -- 'Next Day Air', 'Slow' if 'Standard','Other' for all else including NULL. Then count how many customers fall into each category.
144 -- Expected Columns: Delivery Speed, Customer Count
145
146
147 SELECT COUNT(*) AS Customer_Count,
148   CASE
149     WHEN shipping_type='Express' OR shipping_type='Next Day Air' THEN 'FAST'
150     WHEN shipping_type='Standard' THEN 'Slow'
151     ELSE 'Null'
152   END AS Delivery_Speed,
153   FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
154   GROUP BY Delivery_Speed;
```

↳ Results ▾ Chart

# CUSTOMER_COUNT	A DELIVERY_SPEED
1	89 FAST
2	166 Null
3	45 Slow

-- Q18.Find customers whose purchase_amount is NULL and whose Promo Code Used is 'Yes'.

-- Expected Columns: Customer ID, purchase_amount, Promo Code Used

The screenshot shows a code editor with a SQL query. The code includes comments for Q18 and Q19, and expected columns for both. The results table has three columns: CUSTOMER_ID, PURCHASE_AMOUNT, and PROMO_CODE_USED. All rows show CUSTOMER_ID values (1-5), PURCHASE_AMOUNT values (129, 138, 225, 298, 30), and PROMO_CODE_USED values (all TRUE). The results table is as follows:

CUSTOMER_ID	PURCHASE_AMOUNT	PROMO_CODE_USED
1	129	TRUE
2	138	TRUE
3	225	TRUE
4	298	TRUE
5	30	TRUE

-- Q19.Group by Location and show the maximum PreviousPurchases, replacing NULLs with 0, only where the average rating is above 4.0.

-- Expected Columns: Location, Max Previous Purchases, Average Review Rating

The screenshot shows a code editor with a SQL query for Q19. The results table has three columns: LOCATION, AVERAGE_REVIRE_RATING, and MAXIMUM_PREVIOUS_PURCHASES. The table shows one row with LOCATION as 'ACCOUNTADMIN', AVERAGE_REVIRE_RATING as 0.0, and MAXIMUM_PREVIOUS_PURCHASES as 0. A note at the bottom says 'Query produced no results'.

LOCATION	AVERAGE_REVIRE_RATING	MAXIMUM_PREVIOUS_PURCHASES
ACCOUNTADMIN	0.0	0

-- Q20.Show customers who have a NULL Shipping Type but made a purchase in the range of 30 to 70 USD.

-- Expected Columns: Customer ID, Shipping Type, purchase_amount, Item Purchased

The screenshot shows a database query editor interface. The top bar includes account information (ACCOUNTADMIN), compute type (COMPUTE_WH (Medium)), share options, and navigation links (Open in Workspaces, Code Versions, Search). The code area contains a SQL script with comments and numbered lines:

```
182
183 -- Q20.Show customers who have a NULL Shipping Type but made a purchase in
184 -- the range of 30 to 70 USD.
185 -- Expected Columns: Customer ID, Shipping Type, purchase_amount, Item Purchased
186
187
188 -- Q20.Show customers who have a NULL Shipping Type but made a purchase in the range of 30 to 70 USD.
189 -- Expected Columns: Customer ID, Shipping Type, purchase_amount, Item Purchased
190
191 SELECT customer_id,
192     shipping_type,
193     purchase_amount,
194     item_purchased
195 FROM PRACTICAL3.COOLESTDrip.SHOPPING_TRENDS
196 WHERE shipping_type IS NULL AND purchase_amount BETWEEN 30 and 70
197 GROUP BY customer_id,shipping_type,purchase_amount,item_purchased;
```

The results tab is selected, displaying the output of the query:

#	CUSTOMER_ID	SHIPPING_TYPE	PURCHASE_AMOUNT	ITEM_PURCHASED
1	15	null	54.0	Jeans
2	105	null	43.0	Shirt
3	196	null	66.0	Coat
4	235	null	38.0	Sandals
5	141	null	37.0	Shorts