

1.

The screenshot shows a SQL IDE interface with a query editor and a results table. The query is as follows:

```

--Practical 2.1 - Advanced SQL
--1. Find all records where size is missing and the purchase_amount is greater than 50. Expected Columns: Customer_ID, size, purchase_amount, Item_Purchased

SELECT *
FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
LIMIT 1)

SELECT Customer_ID, size, purchase_amount, Item_Purchased
FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
WHERE size IS NULL AND purchase_amount > 50;

```

The results table displays the following data:

CUSTOMER_ID	size	PURCHASE_AMOUNT	ITEM_PURCHASED
1	11	74.0	Hoodie
2	18	34.0	Jeans
3	22	89.0	Shirts
4	22	54.0	Shoes
5	42	37.0	Shoes
6	73	63.0	Sandals
7	81	34.0	Shirts
8	81	34.0	Shoes
9	109	51.0	Shirts
10	160	44.0	Coat
11	172	94.0	Sandals
12	179	74.0	Shirts

2.

The screenshot shows a SQL IDE interface with a query editor and a results table. The query is as follows:

```

--2. List the total number of purchases grouped by season, treating null values as "UNKNOWN Season"

SELECT
  COALESCE(SEASON, 'UNKNOWN Season') AS SEASON,
  COUNT(*) AS TOTAL_PURCHASES
FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
GROUP BY SEASON;

```

The results table displays the following data:

SEASON	TOTAL_PURCHASES
Autumn	65
Winter	40
Fall	55
UNKNOWN Season	27
Spring	73

3.

2025-10-16 8:42pm 2025-10-16 8:52pm 2025-10-16 9:13pm 2025-10-16 9:33pm + -

Database Worksheets

2025-10-24 9:18am
2025-10-26 9:10pm
2025-10-23 10:13pm
2025-10-16 9:23pm
2025-10-16 9:42pm

SHOPPING_TRENDS.SHOPPING_DATA Settings Open in Workspace

```

39 --3. List the total number of purchases grouped by season, treating NULL values as 'Unknown Season'.
40 SELECT
41     COALESCE(SEASON, 'Unknown Season') AS SEASON,
42     COUNT(*) AS TOTAL_PURCHASES
43 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
44 GROUP BY SEASON;
45
46 --4. Count how many customers used each Payment Method, treating NULLs as 'Not Provided'.
47 SELECT
48     COALESCE(PAYMENT_METHOD, 'Not Provided') AS PAYMENT_METHOD,
49     COUNT(DISTINCT CUSTOMER_ID) AS CUSTOMER_COUNT
50 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
51 GROUP BY PAYMENT_METHOD;
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

Results Chart

PAYMENT_METHOD	CUSTOMER_COUNT
PayPal	34
Bank Transfer	36
Cash Card	42
Wells	52
Not Provided	30
Cash	40
Credit Card	44

4.

```

54
55
56 --4. Show customers where Promo Code Used is NULL and Review Rating is below 3.0.
57 SELECT CUSTOMER_ID, PROMO_CODE_USED, REVIEW_RATING, ITEM_PURCHASED
58 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
59 WHERE PROMO_CODE_USED IS NULL AND REVIEW_RATING < 3.0;
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

Results Chart

CUSTOMER_ID	PROMO_CODE_USED	REVIEW_RATING	ITEM_PURCHASED
21	NULL	2.5	Jeans
38	NULL	2.6	Jeans
61	NULL	2.5	Jeans
80	NULL	2.6	Sneakers
125	NULL	2.6	Sneakers
128	NULL	2.5	Shoes
180	NULL	2.5	Shorts
285	NULL	2.9	Blouse

5.

```

41
42 --5. Group customers by Shipping Type, and return the average purchase amount, treating missing values as 0.
43 SELECT
44     SHIPPING_TYPE,
45     AVG(COALESCE(PURCHASE_AMOUNT, 0)) AS AVERAGE_PURCHASE_AMOUNT
46 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
47 GROUP BY SHIPPING_TYPE;
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75

```

Results Chart

SHIPPING_TYPE	AVERAGE_PURCHASE_AMOUNT
Standard	47.5666667
Express	53.4545455
Store Pickup	55.3333333
NULL	62.7037037
Free Shipping	30.2142857
Next Day Air	54.8888889
2-Day Shipping	51.5578923

6.

```

87
88
89 -----
90 --6. Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method. Expected
91 Columns: Location, Total Purchases
92
93 SELECT
94     LOCATION,
95     COUNT(*) AS TOTAL_PURCHASES
96 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
97 WHERE PAYMENT_METHOD IS NOT NULL
98 GROUP BY LOCATION
99 HAVING COUNT(*) > 5;
100
101

```

LOCATION	TOTAL_PURCHASES
Maine	41
Kentucky	30
Ill	24
New York	31
Oregon	30
Rhode Island	29
Florida	32
Massachusetts	31
Texas	22

7.

```

102
103 -----
104 --7. Create a column Spender Category that classifies customers using CASE: 'High' if amount > 80, 'Medium' if BETWEEN 50 AND 80,
105 'Low' otherwise. Replace NULLs in purchase_amount with 0. Expected Columns: Customer ID, purchase_amount, Spender Category
106
107 SELECT
108     CUSTOMER_ID,
109     COALESCE(PURCHASE_AMOUNT, 0) AS PURCHASE_AMOUNT,
110     CASE
111         WHEN COALESCE(PURCHASE_AMOUNT, 0) > 80 THEN 'High'
112         WHEN COALESCE(PURCHASE_AMOUNT, 0) BETWEEN 50 AND 80 THEN 'Medium'
113         ELSE 'Low'
114     END AS SPENDER_CATEGORY
115 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS";
116
117

```

CUSTOMER_ID	PURCHASE_AMOUNT	SPENDER_CATEGORY
1	1	Low
2	2	Low
3	3	Low
4	4	Low
5	5	Low
6	6	Medium
7	7	Medium
8	8	Low
9	9	High
10	10	Low

```

74 -----
75 --8. Find customers who have no Previous Purchases value but whose Color is not NULL. Expected Columns: Customer ID, Color, Previous
Purchases
76 SELECT CUSTOMER_ID, COLOR, PREVIOUS_PURCHASES
77 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
78 WHERE PREVIOUS_PURCHASES IS NULL AND COLOR IS NOT NULL;
79
80 -----
81

```

	CUSTOMER_ID	COLOR	PREVIOUS_PURCHASES
1	8	Green	NULL
2	21	Yellow	NULL
3	23	White	NULL
4	37	Maroon	NULL
5	40	Gray	NULL
6	43	Black	NULL
7	44	Green	NULL
8	70	White	NULL
9	73	Maroon	NULL
10	74	Black	NULL

8.

9.

```

80 -----
81
82 --9. 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
83 SELECT
84 COALESCE(FREQUENCY_OF_PURCHASES, 'Unknown') AS FREQUENCY_OF_PURCHASES,
85 SUM(COALESCE(PURCHASE_AMOUNT, 0)) AS TOTAL_PURCHASE_AMOUNT
86 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
87 GROUP BY FREQUENCY_OF_PURCHASES;
88
89 -----
90

```

	FREQUENCY_OF_PURCHASES	TOTAL_PURCHASE_AMOUNT
1	Every 3 Months	1748.0
2	Weekly	2184.0
3	Bi-Weekly	2099.0
4	Monthly	1780.0
5	Unknown	1518.0
6	Fortnightly	2033.0
7	Annually	1785.0
8	Quarterly	2541.0

10.

```

90
91 --10. Display a list of all Category values with the number of times each was purchased, excluding rows where Category is NULL.
    Expected Columns: Category, Total Purchases
92 SELECT
93     CATEGORY,
94     COUNT(*) AS TOTAL_PURCHASES
95 FROM "SHOPPING_TRENDS"."SHOPPING_DATA","ANALYTICS"
96 WHERE CATEGORY IS NOT NULL
97 GROUP BY CATEGORY;
98

```

CATEGORY	TOTAL_PURCHASES
Outerwear	60
Footwear	70
Clothing	50
Accessories	78

11.

```

99
100
101 --11. Return the top 5 locations with the highest total purchase amount, replacing NULLs in amount with 0. Expected Columns:
    Location, Total purchase amount
102 SELECT
103     LOCATION,
104     SUM(COALESCE(PURCHASE_AMOUNT, 0)) AS TOTAL_PURCHASE_AMOUNT
105 FROM "SHOPPING_TRENDS"."SHOPPING_DATA","ANALYTICS"
106 GROUP BY LOCATION
107 ORDER BY TOTAL_PURCHASE_AMOUNT DESC
108 LIMIT 5;
109
110

```

LOCATION	TOTAL_PURCHASE_AMOUNT
Maine	2284.0
Florida	1980.0
Massachusetts	1889.0
Rhode Island	1878.0
Kentucky	1798.0

12.

```

111
112 --12. Group customers by Gender and Size, and count how many entries have a NULL Color. Expected Columns: Gender, Size, Null Color
    Count
113 SELECT
114     GENDER,
115     SIZE,
116     COUNT(*) AS NULL_COLOR_COUNT
117 FROM "SHOPPING_TRENDS"."SHOPPING_DATA","ANALYTICS"
118 WHERE COLOR IS NULL
119 GROUP BY GENDER, SIZE;
120
121

```

GENDER	SIZE	NULL_COLOR_COUNT
Male	NULL	8
Male	M	7
Male	L	6
Male	S	5
Male	NL	5

13.

```

120
121
122 --13. Identify all Item Purchased where more than 3 purchases had NULL Shipping Type. Expected Columns: Item Purchased, NULL
Shipping Type Count
123
124 SELECT
125     ITEM_PURCHASED,
126     COUNT(*) AS NULL_SHIPPING_TYPE_COUNT
127 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
128 WHERE SHIPPING_TYPE IS NULL
129 GROUP BY ITEM_PURCHASED
130 HAVING COUNT(*) > 3;
131

```

Results Chart

	ITEM_PURCHASED	NULL_SHIPPING_TYPE_COUNT
1	Null	4
2	Shirt	5
3	Shoes	4

14.

```

132
133 --14. Show a count of how many customers per Payment Method have NULL Review Rating. Expected Columns: Payment Method, Missing
Review Rating Count
134
135 SELECT
136     PAYMENT_METHOD,
137     COUNT(*) AS MISSING_REVIEW_RATING_COUNT
138 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
139 WHERE REVIEW_RATING IS NULL
140 GROUP BY PAYMENT_METHOD;
141

```

Results Chart

	PAYMENT_METHOD	MISSING_REVIEW_RATING_COUNT
1	Credit Card	8
2	Cash	4
3	Null	2
4	Debit Card	7
5	Venmo	9
6	PayPal	3
7	Bank Transfer	4

15.

```

142
143 --15. Group by Category and return the average Review Rating, replacing NAs with 0, and filter only where average is greater than
3.5. Expected Columns: Category, Average Review Rating
144
145 SELECT
146     CATEGORY,
147     AVG(COALESCE(REVIEW_RATING, 0)) AS AVERAGE_REVIEW_RATING
148 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
149 GROUP BY CATEGORY
150 HAVING AVG(COALESCE(REVIEW_RATING, 0)) > 3.5;
151

```

Results Chart

CATEGORY	AVERAGE_REVIEW_RATING
Query produced no results	

16.

```
151
152 -----
153 --16. List all Colors that are missing (NULL) in at least 2 rows and the average Age of customers for those rows. Expected Columns:
154 Color, Average Age
155
156 SELECT
157     COLOR,
158     AVG(AGE) AS AVERAGE_AGE
159 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
160 WHERE COLOR IS NULL
161 GROUP BY COLOR
162 HAVING COUNT(*) >= 2;
```

Results Chart

COLOR	AVERAGE_AGE
1 null	47.8661038

17.

```
161
162 -----
163 --17. Use CASE to create a column Delivery Speed: 'Fast' if Shipping Type is 'Express' or 'Next Day Air', 'Slow' if 'Standard',
164 'Other' for all else including MAIL. Then count how many customers fall into each category. Expected Columns: Delivery Speed,
165 Customer Count
166
167 SELECT
168     CASE
169         WHEN SHIPPING_TYPE IN ('Express', 'Next Day Air') THEN 'Fast'
170         WHEN SHIPPING_TYPE = 'Standard' THEN 'Slow'
171         ELSE 'Other'
172     END AS DELIVERY_SPEED,
173     COUNT(DISTINCT CUSTOMER_ID) AS CUSTOMER_COUNT
174 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
175 GROUP BY DELIVERY_SPEED;
```

Results Chart

DELIVERY_SPEED	CUSTOMER_COUNT
1 Fast	68
2 Slow	45
3 Other	166

18.

```
174
175 -----
176 --18. Find customers whose purchase_amount is NULL and whose Promo Code Used is "Yes". Expected Columns: Customer ID,
177 purchase_amount, Promo Code Used
178
179 SELECT CUSTOMER_ID, PURCHASE_AMOUNT, PROMO_CODE_USED
180 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
181 WHERE PURCHASE_AMOUNT IS NULL AND PROMO_CODE_USED = "Yes";
```

Results Chart

CUSTOMER_ID	PURCHASE_AMOUNT	PROMO_CODE_USED
1	13	TRUE
2	30	TRUE
3	78	TRUE
4	95	TRUE
5	124	TRUE
6	129	TRUE
7	130	TRUE
8	138	TRUE
9	153	TRUE
10	168	TRUE

19.

```

181
182
183 --19. Group by Location and show the maximum Previous Purchases, replacing NULLs with 0, only where the average rating is above 4.0.
Expected Columns: Location, Max Previous Purchases, Average Review Rating
184
185 SELECT
186     LOCATION,
187     MAX(COALESCE(PREVIOUS_PURCHASES, 0)) AS MAX_PREVIOUS_PURCHASES,
188     AVG(REVIEW_RATING) AS AVERAGE_REVIEW_RATING
189 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
190 GROUP BY LOCATION
191 HAVING AVG(REVIEW_RATING) > 4.0;
192
193

```

Results Chart

LOCATION	MAX_PREVIOUS_PURCHASES	AVERAGE_REVIEW_RATING
Query produced no results.		

20.

```

194
195
196 --20. 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
197
198 SELECT CUSTOMER_ID, SHIPPING_TYPE, PURCHASE_AMOUNT, ITEM_PURCHASED
199 FROM "SHOPPING_TRENDS"."SHOPPING_DATA"."ANALYTICS"
200 WHERE SHIPPING_TYPE IS NULL
201 AND PURCHASE_AMOUNT BETWEEN 30 AND 70;
202
203
204

```

Results Chart

#	CUSTOMER_ID	SHIPPING_TYPE	PURCHASE_AMOUNT	ITEM_PURCHASED
1	13	NULL	54.0	Jeans
2	105	NULL	43.0	Shirt
3	141	NULL	37.0	Shorts
4	188	NULL	66.0	Coat
5	213	NULL	36.0	Shirt
6	225	NULL	50.0	Sandals
7	293	NULL	35.0	NULL