

1.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
8  --Find all records where Size is missing and the purchase_amount is greater than 50--
9  SELECT
10    customer_id,
11    size,
12    purchase_amount,
13    item_purchased
14   FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
15   WHERE size IS NULL
16     AND purchase_amount > 50;
17
```

↳ Results ▾ Chart

	# 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
6		73 null		65.0 Sandals
7		91 null		54.0 Shoes
8		97 null		56.0 Shoes
9		100 null		55.0 Sneakers
10		160 null		84.0 Coat
11		173 null		96.0 Sandals
...	

Activate
Go to Setti

2.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
--  
--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 "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
GROUP BY COALESCE(season, 'Unknown Season');
```

↳ Results ↵ Chart

	SEASON	TOTAL_PURCHASES
1	Summer	65
2	Winter	80
3	Fall	55
4	Spring	73
5	Unknown Season	27

3.

The screenshot shows a database interface with a query editor and a results table. The query editor displays the following SQL code:

```
23
24
25 --Count how many customers used each Payment Method, treating NULLs as 'Not Provided'--
26
27 SELECT
28   COALESCE(payment_method, 'Not Provided') AS Payment_Method,
29   COUNT(DISTINCT customer_id) AS Customer_Count
30
31   FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
32   GROUP BY COALESCE(payment_method, 'Not Provided');
```

The results table has two columns: PAYMENT_METHOD and CUSTOMER_COUNT. The data is as follows:

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

At the bottom right of the results table, there is a button labeled "Activate V".

4.

The screenshot shows a database interface with the following details:

Top right corner: ACCOUNTADMIN • COM

Database: PRACTICAL3.SHOPPINGTRENDS

Query:

```
31
32 --Show customers where Promo Code Used is NULL and Review Rating is below 3.0--
33
34     SELECT
35         customer_id,
36         promo_code_used,
37         review_rating,
38         item_purchased
39     FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
40     WHERE promo_code_used IS NULL
41     AND review_rating < 3.0;
```

Results tab is selected.

#	CUSTOMER_ID	PROMO_CODE_USED	REVIEW_RATING	ITEM_PURCHASED
1	21	null	2.5	Jeans
2	38	null	2.6	Jeans
3	61	null	2.5	Jeans
4	80	null	2.6	Sneakers
5	125	null	2.8	Sneakers
6	128	null	2.5	Shoes
7	180	null	2.5	Shorts
8	285	null	2.9	Blouse

5.

The screenshot shows a database interface with a code editor and a results table. The code editor contains a SQL query:

```
41 --Group customers by Shipping Type, and return the average purchase_amount, treating missing values as 0--  
42  
43     SELECT  
44         COALESCE(shipping_type, 'Unknown') AS Shipping_Type,  
45         AVG(COALESCE(purchase_amount, 0)) AS Average_Purchase_Amount  
46     FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
47     GROUP BY COALESCE(shipping_type, 'Unknown');  
48  
49  
50
```

The results table has two columns: SHIPPING_TYPE and AVERAGE_PURCHASE_AMOUNT. The data is as follows:

SHIPPING_TYPE	AVERAGE_PURCHASE_AMOUNT
Standard	47.6666667
Express	53.4545455
Store Pickup	55.3333333
Free Shipping	50.2142857
Next Day Air	54.8666667
Unknown	52.7037037
2-Day Shipping	51.5576923

6.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
49 --Display the number of purchases per Location only for those with more than 5 purchases and no NULL Payment Method--
50 SELECT
51     location,
52     COUNT(*) AS Total_Purchases
53 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
54 WHERE payment_method IS NOT NULL
55 GROUP BY location
56 HAVING COUNT(*) > 5;
57
58
```

↳ Results ↵ Chart

	LOCATION	TOTAL_PURCHASES
1	Maine	41
2	Kentucky	30
3	null	24
4	New York	31
5	Oregon	30
6	Rhode Island	29
7	Florida	32
8	Massachusetts	31
9	Texas	22

7.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
58 --Create a column Spender Category that classifies customers using CASE--
59 SELECT
60   customer_id,
61   COALESCE(purchase_amount, 0) AS purchase_amount,
62   CASE
63     WHEN COALESCE(purchase_amount, 0) > 80 THEN 'High'
64     WHEN COALESCE(purchase_amount, 0) BETWEEN 50 AND 80 THEN 'Medium'
65     ELSE 'Low'
66   END AS Spender_Category
67 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA";
68
```

↳ Results ▾ Chart

#	CUSTOMER_ID	# PURCHASE_AMOUNT	▲ SPENDER_CATEGORY
1	1	20.0	Low
2	2	21.0	Low
3	3	27.0	Low
4	4	45.0	Low
5	5	80.0	Medium
6	6	82.0	High
7	7	50.0	Medium
8	8	29.0	Low
9	9	100.0	High
10	10	97.0	High
11	11	74.0	Medium
...

Activate View
Go to Settings

8.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
68
69  --Find customers who have no Previous Purchases value but whose Color is not NULL--
70  SELECT
71      customer_id,
72      color,
73      previous_purchases
> 74  FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
75  WHERE previous_purchases IS NULL
76      AND color IS NOT NULL;
77
78
```

↳ Results ⚡ Chart

#	CUSTOMER_ID	▲ COLOR	#	PREVIOUS_PURCHASES
1		8	Green	null
2		21	Yellow	null
3		25	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		75	Pink	null
11		83	Black	null
...	

Activate Go to Settings

9.

The screenshot shows a database interface with a query editor and a results table. The query editor contains the following SQL code:

```
--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(purchase_amount) AS Total_purchase_amount  
FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
GROUP BY COALESCE(frequency_of_purchases, 'Unknown');
```

The results table has two columns: FREQUENCY_OF_PURCHASES and TOTAL_PURCHASE_AMOUNT. The data is as follows:

	FREQUENCY_OF_PURCHASES	TOTAL_PURCHASE_AMOUNT
1	Every 3 Months	1749.0
2	Weekly	2184.0
3	Bi-Weekly	2099.0
4	Monthly	1780.0
5	Fortnightly	2033.0
6	Annually	1765.0
7	Unknown	1518.0
8	Quarterly	2541.0

10.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
-- Display a list of all Category values with the number of times each was purchased, excluding rows where Category is NULL--  
85  SELECT  
86      category,  
87      COUNT(*) AS Total_Purchases  
88  FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
89  WHERE category IS NOT NULL  
90  GROUP BY category;  
91  
92  
93
```

↳ Results ▾ Chart

CATEGORY	TOTAL_PURCHASES
1 Outerwear	60
2 Footwear	70
3 Clothing	59
4 Accessories	78

11.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
93 --Return the top 5 Locations with the highest total purchase_amount, replacing NULLs in amount with 0--
94 SELECT
95   location,
96   SUM(COALESCE(purchase_amount, 0)) AS Total_purchase_amount
97 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
98 GROUP BY location
99 ORDER BY Total_purchase_amount DESC
100 LIMIT 5;
101
102
```

↳ Results  Chart

	LOCATION	TOTAL_PURCHASE_AMOUNT
1	Maine	2294.0
2	Florida	1980.0
3	Massachusetts	1899.0
4	Rhode Island	1876.0
5	Kentucky	1798.0

12.

The screenshot shows a database interface with a query editor and a results table. The query editor contains the following SQL code:

```
110
111 --Identify all Item Purchased where more than 3 purchases had NULL Shipping Type--
112 SELECT
113     item_purchased,
114     COUNT(*) AS Null_Shipping_Type_Count
115 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
116 WHERE shipping_type IS NULL
117 GROUP BY item_purchased
118 HAVING COUNT(*) > 3;
119
```

The results table has two columns: ITEM_PURCHASED and NULL_SHIPPING_TYPE_COUNT. The data is as follows:

ITEM_PURCHASED	NULL_SHIPPING_TYPE_COUNT
null	4
Shirt	5
Shoes	4

13.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
119
120 --Identify all Item Purchased where more than 3 purchases had NULL Shipping Type--
121 SELECT
122     item_purchased,
123     COUNT(*) AS Null_Shipping_Type_Count
124 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
125 WHERE shipping_type IS NULL
126 GROUP BY item_purchased
127 HAVING COUNT(*) > 3;
```

↳ Results ↵ Chart

ITEM_PURCHASED	NULL_SHIPPING_TYPE_COUNT
1 null	4
2 Shirt	5
3 Shoes	4

14.

The screenshot shows a database interface with a code editor and a results table. The code editor contains a SQL query to count customers by payment method who have a NULL review rating. The results table displays the count for each payment method.

```
PRACTICAL3.SHOPPINGTRENDS < Settings <

128
129 --Show a count of how many customers per Payment Method have NULL Review Rating--
130
131     COALESCE(payment_method, 'Not Provided') AS Payment_Method,
132     COUNT(*) AS Missing_Review_Rating_Count
133
134     FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
135     WHERE review_rating IS NULL
136     GROUP BY COALESCE(payment_method, 'Not Provided');
```

Results

PAYMENT_METHOD	MISSING REVIEW RATING COUNT
Credit Card	8
Cash	4
Debit Card	7
Not Provided	2
Venmo	9
PayPal	3
Bank Transfer	4

15.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
137 --Group by Category and return the average Review Rating, replacing NULLs with 0, and filter only where average is greater than 3.5--  
138 SELECT  
139     category,  
140     AVG(COALESCE(review_rating, 0)) AS Average_Review_Rating  
141 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
142 GROUP BY category  
143 HAVING AVG(COALESCE(review_rating, 0)) > 3.5;  
144  
145
```

↳ Results ↳ Chart

CATEGORY	AVERAGE REVIEW RATING
Query produced no results	

16.

The screenshot shows a database interface with a query editor and a results table. The top bar includes account information: ACCOUNTADMIN and COMI. The query editor displays the following SQL code:

```
145 --List all Colors that are missing (NULL) in at least 2 rows and the average Age of customers for those rows--
146 SELECT
147     color,
148     AVG(age) AS Average_Age
149 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
150 WHERE color IS NULL
151 GROUP BY color
152 HAVING COUNT(*) >= 2;
153
```

The results table has two columns: COLOR and AVERAGE_AGE. The single row shows 'null' in the COLOR column and '47.8461538' in the AVERAGE_AGE column.

COLOR	AVERAGE_AGE
null	47.8461538

17.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
--  
154 --Use CASE to create a column Delivery Speed--  
155 SELECT  
156     CASE  
157         WHEN shipping_type IN ('Express', 'Next Day Air') THEN 'Fast'  
158         WHEN shipping_type = 'Standard' THEN 'Slow'  
159         ELSE 'Other'  
160     END AS Delivery_Speed,  
161     COUNT(DISTINCT customer_id) AS Customer_Count  
162 FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
163 GROUP BY  
164     CASE  
165         WHEN shipping_type IN ('Express', 'Next Day Air') THEN 'Fast'  
166         WHEN shipping_type = 'Standard' THEN 'Slow'  
167         ELSE 'Other'  
168     END;
```

↳ Results ▾ Chart

	DELIVERY_SPEED	CUSTOMER_COUNT
1	Fast	89
2	Slow	45
3	Other	166

18.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
170 --Find customers whose purchase_amount is NULL and whose Promo Code Used is 'Yes'--  
171 SELECT  
172     customer_id,  
173     purchase_amount,  
174     promo_code_used  
175    FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
176   WHERE purchase_amount IS NULL  
177     AND promo_code_used = 'Yes';  
178  
179
```

↳ Results ↳ Chart

#	CUSTOMER_ID	#	PURCHASE_AMOUNT	0 1	PROMO_CODE_USED
1			13	null	TRUE
2			30	null	TRUE
3			78	null	TRUE
4			95	null	TRUE
5			124	null	TRUE
6			129	null	TRUE
7			130	null	TRUE
8			138	null	TRUE
9			153	null	TRUE
10			168	null	TRUE
11			177	null	TRUE
12			202	null	TRUE

Activate | Go to Settings

19.

The screenshot shows a database query interface with the following details:

Top right corner: ACCOUNTADMIN CON

Toolbar: PRACTICAL3.SHOPTRENDS ▾ Settings ▾

Query code (lines 179-187):

```
--Group by Location and show the maximum Previous Purchases, replacing NULLs with 0, only where the average rating is above 4.0--  
179  SELECT  
180    location,  
181    MAX(COALESCE(previous_purchases, 0)) AS Max_Previous_Purchases,  
182    AVG(review_rating) AS Average_Review_Rating  
183   FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"  
184   GROUP BY location  
185   HAVING AVG(review_rating) > 4.0;  
186  
187
```

Result tab: Results (selected) ▾ Chart

Table Headers:

LOCATION	MAX_PREVIOUS_PURCHASES	AVERAGE REVIEW RATING
----------	------------------------	-----------------------

Message: Query produced no results

20.

PRACTICAL3.SHOPPINGTRENDS ▾ Settings ▾

```
187 --Show customers who have a NULL Shipping Type but made a purchase in the range of 30 to 70 USD--
188
189 SELECT
190     customer_id,
191     shipping_type,
192     purchase_amount,
193     item_purchased
194     FROM "PRACTICAL3"."SHOPPINGTRENDS"."DATA"
195     WHERE shipping_type IS NULL
196     AND purchase_amount BETWEEN 30 AND 70;
```

↳ Results ↵ Chart

	# CUSTOMER_ID	▲ SHIPPING_TYPE	# PURCHASE_AMOUNT	▲ ITEM_PURCHASED
1		15	54.0	Jeans
2		105	43.0	Shirt
3		141	37.0	Shorts
4		196	66.0	Coat
5		213	36.0	Shirt
6		235	38.0	Sandals
7		293	35.0	null