BrightLight Data Analytics Coding Practical

Practical 2.1: Advanced SQL

The following questions are designed to help you build a strong foundation in basic SQL syntax. You are provided with a dataset named shoping_trends.csv. Upload this dataset to your Snowflake account and use it to answer the questions below.

Please follow the instructions below carefully:

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*

PRACTICAL_1.PUBLIC Settings

```sql
SELECT
  customer_id,
  size,
  purchase_amount,
  item_purchased
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE size IS NULL AND purchase_amount > 50;
```

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

2. List the total number of purchases grouped by Season, treating NULL values as 'Unknown Season'.
   Expected Columns: Season, Total Purchases

```sql
--Total purchases by Season, using 'Unknown Season' for NULL
SELECT
  IFNULL(season, 'Unknown Season') AS season,
  COUNT(*) AS total_purchases
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY COALESCE(season, 'Unknown Season');
```

Results Chart

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

3. Count how many customers used each Payment Method, treating NULLs as 'Not Provided'.
   Expected Columns: Payment Method, Customer Count

```sql
--Count customers by Payment Method, treating NULL as 'Not Provided'
SELECT
  IFNULL(payment_method, 'Not Provided') AS payment_method,
  COUNT(*) AS customer_count
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY COALESCE(payment_method, 'Not Provided');
```

Results | Chart

	PAYMENT_METHOD	CUSTOMER_COUNT
1	PayPal	51
2	Not Provided	30
3	Credit Card	44
4	Venmo	53
5	Debit Card	42

4. 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

```sql
--Customers with NULL Promo Code Used and Review Rating below 3.0
SELECT
  customer_id,
  promo_code_used,
  review_rating,
  item_purchased
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE promo_code_used IS NULL AND review_rating < 3.0;
```

Results | Chart

	CUSTOMER_ID	0|1 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

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

```sql
--Average purchase_amount by Shipping Type, treating NULL as 0
SELECT
  shipping_type,
  AVG(IFNULL(purchase_amount, 0)) AS average_purchase_amount
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY shipping_type;
```

Results | Chart

	SHIPPING_TYPE	AVERAGE_PURCHASE_AMOUNT
1	Free Shipping	50.2142857
2	Store Pickup	55.3333333
3	null	52.7037037
4	Express	53.4545455
5	Standard	47.6666667

6. 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

```sql
--Locations with more than 5 purchases and no NULL Payment Method
SELECT
  location,
  COUNT(*) AS total_purchases
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE payment_method IS NOT NULL
GROUP BY location
HAVING COUNT(*) > 5;
```

Results | Chart

	LOCATION	TOTAL_PURCHASES
1	null	24
2	Maine	41
3	Oregon	30
4	Kentucky	30
5	Florida	32

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.
   Expected Columns: Customer ID, purchase_amount, Spender Category

```sql
SELECT
  customer_id,
  IFNULL(purchase_amount, 0) AS purchase_amount,
  CASE
    WHEN IFNULL(purchase_amount, 0) > 80 THEN 'High'
    WHEN IFNULL(purchase_amount, 0) BETWEEN 50 AND 80 THEN 'Medium'
    ELSE 'Low'
  END AS spender_category
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS";
```

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

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

```sql
--Customers with NULL Previous Purchases but non-NULL Color
SELECT
  customer_id,
  color,
  previous_purchases
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE previous_purchases IS NULL AND color IS NOT NULL;
```

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

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

```sql
--Total spent per Frequency of Purchases (NULL = 'Unknown')
SELECT
  IFNULL(frequency_of_purchases, 'Unknown') AS frequency,
  SUM(purchase_amount) AS total_spent
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY COALESCE(frequency_of_purchases, 'Unknown');
```

Results | Chart

	FREQUENCY	TOTAL_SPENT
1	Annually	1765.0
2	Monthly	1780.0
3	Bi-Weekly	2099.0
4	Quarterly	2541.0
5	Every 3 Months	1749.0

10. Display a list of all Category values with the number of times each was purcha sed, excluding rows where Categoryis NULL.
    Expected Columns: Category, Total Purchases

```sql
--Count how many times each Category was purchased (exclude NULLs)
SELECT
  category,
  COUNT(*) AS total_purchases
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE category IS NOT NULL
GROUP BY category;
```

Results | Chart

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

11. Return the top 5 Locations with the highest total purchase_amount, replacing NULLs in amount with 0.
    Expected Columns: Location, Total purchase_amount

```sql
--Top 5 Locations by total purchase_amount (treat NULLs as 0)
SELECT
  location,
  SUM(IFNULL(purchase_amount, 0)) AS total_purchase_amount
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY location
ORDER BY total_purchase_amount DESC
LIMIT 5;
```

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. Group customers by Gender and Size, and count how many entries have a NULL Color.
    Expected Columns: Gender, Size, Null Color Count

```sql
--Count of NULL Colors grouped by Gender and Size
SELECT
  gender,
  size,
  COUNT(*) AS null_color_count
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE color IS NULL
GROUP BY gender, size;
```

Results | Chart

	GENDER	SIZE	NULL_COLOR_COUNT
1	Male	S	5
2	Male	null	6
3	Male	L	6
4	Male	M	7
5	Male	XL	5

13. Identify all Item Purchased where more than 3 purchases had NULL Shipping Type.
    Expected Columns: Item Purchased, NULL Shipping Type Count

```sql
--Items Purchased with more than 3 NULL Shipping Types
SELECT
  item_purchased,
  COUNT(*) AS null_shipping_type_count
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE shipping_type IS NULL
GROUP BY item_purchased
HAVING COUNT(*) > 3;
```

Results | Chart

	ITEM_PURCHASED	NULL_SHIPPING_TYPE_COUNT
1	Shirt	5
2	null	4
3	Shoes	4

14. Show a count of how many customers per Payment Method have NULL Review Rating.
    Expected Columns: Payment Method, Missing Review Rating Count

```sql
--Count of NULL Review Ratings per Payment Method
SELECT
  payment_method,
  COUNT(*) AS missing_review_rating_count
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE review_rating IS NULL
GROUP BY payment_method;
```

Results | Chart

	PAYMENT_METHOD	MISSING_REVIEW_RATING_COUNT
1	PayPal	3
2	null	2
3	Credit Card	8
4	Venmo	9
5	Cash	4

15. 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

```sql
--Average Review Rating by Category (NULL = 0, filter > 3.5)
SELECT
  category,
  AVG(IFNULL(review_rating, 0)) AS average_review_rating
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY category
HAVING AVG(IFNULL(review_rating, 0)) > 3.5;
```

Results | Chart

CATEGORY	AVERAGE_REVIEW_RATING

Query produced no results

16. 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

```sql
--Colors missing in at least 2 rows and avg Age for those rows
SELECT
  color,
  AVG(age) AS average_age
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE color IS NULL
GROUP BY color
HAVING COUNT(*) >= 2;
```

Results | Chart

	COLOR	AVERAGE_AGE
1	null	47.8461538

17. 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

```sql
--Delivery Speed column using CASE + count per type
SELECT
  CASE
    WHEN shipping_type IN ('Express', 'Next Day Air') THEN 'Fast'
    WHEN shipping_type = 'Standard' THEN 'Slow'
    ELSE 'Other'
  END AS delivery_speed,
  COUNT(*) AS customer_count
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
GROUP BY delivery_speed;
```

Results | Chart

	DELIVERY_SPEED	CUSTOMER_COUNT
1	Slow	45
2	Fast	89
3	Other	166

18. Find customers whose purchase_amount is NULL and whose Promo Code Used is 'Yes'.
    Expected Columns: Customer ID, purchase_amount, Promo Code Used

```sql
-- Customers where purchase_amount is NULL and Promo Code Used = 'Yes'
SELECT
  customer_id,
  purchase_amount,
  promo_code_used
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE purchase_amount IS NULL AND promo_code_used = 'Yes';
```

Results | Chart

	CUSTOMER_ID	PURCHASE_AMOUNT	PROMO_CODE_USED
1	13	null	TRUE
2	30	null	TRUE
3	78	null	TRUE
4	95	null	TRUE

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

```sql
--Max Previous Purchases per Location (NULL = 0), average rating > 4.0
SELECT
  location,
  MAX(IFNULL(previous_purchases, 0)) AS max_previous_purchases,
  AVG(review_rating) AS average_review_rating
FROM shopping_trends
GROUP BY location
HAVING AVG(review_rating) > 4.0;
```

Results | Chart

LOCATION	MAX_PREVIOUS_PURCHASES	AVERAGE_REVIEW_RATING

Query produced no results

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

```sql
--Customers with NULL Shipping Type and purchase_amount between 30 and 70
SELECT
  customer_id,
  shipping_type,
  purchase_amount,
  item_purchased
FROM "PRACTICAL_1"."PUBLIC"."SHOPPING_TRENDS"
WHERE shipping_type IS NULL
  AND purchase_amount BETWEEN 30 AND 70;
```

Results | Chart

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