COMPANY OVERVIEW: Company RFM Segmentation Overview

```
WITH user_purchases AS (
  SELECT
    user_id,
    MAX(date) AS last_purchase_date,
    -- date-last_purchase_date AS recency, MySQL works, however SQLlite doesn't allow this
    COUNT(DISTINCT delivery_id) AS frequency,
    SUM(sales) AS monetary,
    SUM(sales) / COUNT(DISTINCT delivery_id) AS spend_per_delivery
  FROM transaction_data
  WHERE date BETWEEN '2023-06-01' AND '2023-09-30'
  GROUP BY user_id
),

user_purchases_with_lookback AS (
  SELECT
    *,
    (last_purchase_date - INTERVAL '26 weeks') AS lookback_weeks,
    (DATE '2023-09-30' - last_purchase_date) AS recency
  FROM user_purchases
),

order_hist AS (
  SELECT
    user_id,
    COUNT(DISTINCT CASE WHEN date >= lookback_weeks AND date <= last_purchase_date
THEN delivery_id END) AS deliveries_lookback_weeks
  FROM transaction_data
  LEFT JOIN user_purchases_with_lookback USING (user_id)
  WHERE date BETWEEN (DATE '2023-06-01' - INTERVAL '26 weeks') AND '2023-09-30'
  GROUP BY user_id
),

final_agg AS (
  SELECT
    user_purchases_with_lookback.*,
    CASE WHEN order_hist.deliveries_lookback_weeks = 1 THEN 'new_user' ELSE
'returning_user' END AS user_type,
    order_hist.deliveries_lookback_weeks
  FROM user_purchases_with_lookback
  LEFT JOIN order_hist USING (user_id)
),
```

```
weighted_score_agg AS (
  SELECT
    *,
    (PTILE_RECENCY + PTILE_FREQUENCY + ptile_spd) AS weight_score
  FROM (
    SELECT
      user_id,
      recency,
      frequency,
      monetary,
      spend_per_delivery,
      user_type,
      NTILE(3) OVER (PARTITION BY user_type ORDER BY recency DESC) AS ptile_recency,
      NTILE(3) OVER (PARTITION BY user_type ORDER BY frequency) AS ptile_frequency,
      NTILE(3) OVER (PARTITION BY user_type ORDER BY spend_per_delivery) AS ptile_spd
    FROM final_agg
  ) t
)

SELECT
  CASE
    WHEN weight_score >= 8 THEN '1_high'
    WHEN weight_score >= 5 AND weight_score < 8 THEN '2_medium'
    ELSE '3_low'
  END AS segment,
  user_type,
  COUNT(DISTINCT user_id) AS user_count
FROM weighted_score_agg
GROUP BY user_type, segment
ORDER BY segment, user_type;
```

CATEGORY OVERVIEW: Category RFM Segmentation Overview Agg.

```
WITH user_purchases AS (
  SELECT
    user_id,
    category,
    MAX(date) AS last_purchase_date,
    COUNT(DISTINCT delivery_id) AS frequency,
    SUM(sales) AS monetary,
    SUM(sales) / COUNT(DISTINCT delivery_id) AS spend_per_delivery
  FROM transaction_data
  WHERE date BETWEEN '2023-06-01' AND '2023-09-30'
  GROUP BY user_id, category
),

user_purchases_with_lookback AS (
  SELECT
    *,
    (last_purchase_date - INTERVAL '26 weeks') AS lookback_weeks,
    (DATE '2023-09-30' - last_purchase_date) AS recency
  FROM user_purchases
),

order_hist AS (
  SELECT
    user_id,
    category,
    COUNT(DISTINCT CASE WHEN date >= lookback_weeks AND date <= last_purchase_date
THEN delivery_id END) AS deliveries_lookback_weeks
  FROM transaction_data
  LEFT JOIN user_purchases_with_lookback USING (user_id, category)
  WHERE date BETWEEN (DATE '2023-06-01' - INTERVAL '26 weeks') AND '2023-09-30'
  GROUP BY user_id, category
),

final_agg AS (
  SELECT
    user_purchases_with_lookback.*,
    CASE WHEN order_hist.deliveries_lookback_weeks = 1 THEN 'new_user' ELSE
'returning_user' END AS user_type,
    order_hist.deliveries_lookback_weeks
  FROM user_purchases_with_lookback
  LEFT JOIN order_hist USING (user_id, category)
),
```

```
weighted_score_agg AS (
  SELECT
    *,
    (PTILE_RECENCY + PTILE_FREQUENCY + ptile_spd) AS weight_score
  FROM (
    SELECT
      user_id,
      category,
      recency,
      frequency,
      monetary,
      spend_per_delivery,
      user_type,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY recency DESC) AS
ptile_recency,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY frequency) AS
ptile_frequency,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY spend_per_delivery) AS
ptile_spd
    FROM final_agg
  ) t
)

SELECT
  category,
  CASE
    WHEN weight_score >= 8 THEN '1_high'
    WHEN weight_score >= 5 AND weight_score < 8 THEN '2_medium'
    ELSE '3_low'
  END AS segment,
  user_type,
  COUNT(DISTINCT user_id) as user_count
FROM weighted_score_agg
GROUP BY category, segment, user_type
ORDER BY category, segment, user_type;
```

CATEGORY OVERVIEW: Category RFM Segmentation Overview by User Level

```
WITH user_purchases AS (
  SELECT
    user_id,
    category,
    MAX(date) AS last_purchase_date,
    COUNT(DISTINCT delivery_id) AS frequency,
    SUM(sales) AS monetary,
    SUM(sales) / COUNT(DISTINCT delivery_id) AS spend_per_delivery
  FROM transaction_data
  WHERE date BETWEEN '2023-06-01' AND '2023-09-30'
  GROUP BY user_id, category
),

user_purchases_with_lookback AS (
  SELECT
    *,
    (last_purchase_date - INTERVAL '26 weeks') AS lookback_weeks,
    (DATE '2023-09-30' - last_purchase_date) AS recency
  FROM user_purchases
),

order_hist AS (
  SELECT
    user_id,
    category,
    COUNT(DISTINCT CASE WHEN date >= lookback_weeks AND date <= last_purchase_date
THEN delivery_id END) AS deliveries_lookback_weeks
  FROM transaction_data
  LEFT JOIN user_purchases_with_lookback USING (user_id, category)
  WHERE date BETWEEN (DATE '2023-06-01' - INTERVAL '26 weeks') AND '2023-09-30'
  GROUP BY user_id, category
),

final_agg AS (
  SELECT
    user_purchases_with_lookback.*,
    CASE WHEN order_hist.deliveries_lookback_weeks = 1 THEN 'new_user' ELSE
'returning_user' END AS user_type,
    order_hist.deliveries_lookback_weeks
  FROM user_purchases_with_lookback
  LEFT JOIN order_hist USING (user_id, category)
),
```

```
weighted_score_agg AS (
  SELECT
    *,
    (PTILE_RECENCY + PTILE_FREQUENCY + ptile_spd) AS weight_score
  FROM (
    SELECT
      user_id,
      category,
      recency,
      frequency,
      monetary,
      spend_per_delivery,
      user_type,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY recency DESC) AS
ptile_recency,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY frequency) AS
ptile_frequency,
      NTILE(3) OVER (PARTITION BY user_type, category ORDER BY spend_per_delivery) AS
ptile_spd
    FROM final_agg
  ) t
)

SELECT
  user_id,
  category,
  user_type,
  CASE
    WHEN weight_score >= 8 THEN '1_high'
    WHEN weight_score >= 5 AND weight_score < 8 THEN '2_medium'
    ELSE '3_low'
  END AS segment
FROM weighted_score_agg
ORDER BY USER_ID;
```

COMPANY OVERVIEW: USER TARGETING

```
WITH user_purchases AS (
  SELECT
    user_id,
    category,
    MAX(date) AS last_purchase_date,
    COUNT(DISTINCT delivery_id) AS frequency,
    SUM(sales) AS monetary,
    SUM(sales) / COUNT(DISTINCT delivery_id) AS spend_per_delivery
  FROM transaction_data
  WHERE date BETWEEN '2023-06-01' AND '2023-09-30'
  GROUP BY user_id, category
),

user_purchases_with_recency AS (
  SELECT
    *,
    (DATE '2023-09-30' - last_purchase_date) AS recency
  FROM user_purchases
),

weighted_score_agg AS (
  SELECT
    *,
    (ptile_recency + ptile_frequency + ptile_spd) AS weight_score
  FROM (
    SELECT
      user_id,
      category,
      recency,
      frequency,
      monetary,
      spend_per_delivery,
      NTILE(3) OVER (PARTITION BY category ORDER BY recency DESC) AS ptile_recency,
      NTILE(3) OVER (PARTITION BY category ORDER BY frequency) AS ptile_frequency,
      NTILE(3) OVER (PARTITION BY category ORDER BY spend_per_delivery) AS ptile_spd
    FROM user_purchases_with_recency
  ) t
),

user_segment AS (
  SELECT DISTINCT
    user_id,
```

```
    category,
    CASE
      WHEN weight_score >= 8 THEN '1_high'
      WHEN weight_score >= 5 AND weight_score < 8 THEN '2_medium'
      ELSE '3_low'
    END AS segment
  FROM weighted_score_agg
  ORDER BY user_id
),

mapping AS (
  SELECT DISTINCT
    user_id,
    location,
    age,
    gender,
    past_buying_history
  FROM transaction_data
)

SELECT
  category,
  segment,
  age,
  gender,
  location,
  past_buying_history,
  COUNT(DISTINCT user_id) AS user_count
FROM user_segment
JOIN mapping USING (user_id)
GROUP BY category, segment, age, gender, location, past_buying_history;
```