Expedia Partner Journey – SQL Queries

Q1 – Friction Points

Extract customer objections from transcripts using JSON speech data.

```
-- Used single row functions for text and JSON queries to deal
with transcripts, filtering records, aggregation, and sorting


SELECT LOWER(transcript_json ->> 'text') AS phrase, COUNT(*)
FROM partner_calls,
     jsonb_array_elements(transcript_json) AS transcript_json
WHERE transcript_json ->> 'speaker_type' = 'customer'
  AND LOWER(transcript_json ->> 'text') LIKE ANY (
    ARRAY[
      '%too expensive%','%not interested%','%already
using%','%budget%',
      '%complex%','%not useful%','%don''t understand%','%not
understand%',
      '%worst%','%not worth%','%not happy%','%stop%','%hate%'
    ])
GROUP BY phrase
ORDER BY COUNT(*) DESC
LIMIT 10;
```

Q2 – Best Practices by Team

Use team-level medians to impute missing behavioral metrics.

```
-- Created CTE and its output in the main query, statistics,

-- Used aggregate functions, filtering records, aggregation,
and sorting


WITH team_medians AS (
  SELECT team,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_talk_to_listen_ratio) AS median_ratio,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_talk_speed) AS median_speed,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_filler_words) AS median_filler
  FROM partner_calls
```

```
  GROUP BY team
)
SELECT
  pc.team,
  ROUND(AVG(COALESCE(pc.rep_talk_to_listen_ratio,
tm.median_ratio)), 2) AS avg_ratio,
  ROUND(AVG(COALESCE(pc.rep_talk_speed, tm.median_speed)), 2)
AS avg_speed,
  ROUND(AVG(COALESCE(pc.rep_filler_words, tm.median_filler)),
2) AS avg_filler,
  ROUND(SUM(pc.oppy_value_pitched)*100.0 /
NULLIF(SUM(pc.oppy_value_served), 0), 2) AS avg_pitch_rate_pct,
  ROUND(SUM(pc.oppy_value_captured)*100.0 /
NULLIF(SUM(pc.oppy_value_pitched), 0), 2) AS
avg_conversion_rate_pct,
  COUNT(*) AS call_volume
FROM partner_calls pc
JOIN team_medians tm ON pc.team = tm.team
WHERE pc.oppy_value_pitched > 0 AND pc.oppy_value_captured > 0
AND pc.oppy_value_served > 0
GROUP BY pc.team
ORDER BY avg_conversion_rate_pct DESC;
```

Use team-level medians to impute missing behavioral metrics.

```
WITH region_medians AS (
  SELECT account_region,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_talk_to_listen_ratio) AS median_ratio,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_talk_speed) AS median_speed,
         PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY
rep_filler_words) AS median_filler
  FROM partner_calls
  GROUP BY account_region
)
SELECT
  pc.account_region,
  ROUND(AVG(COALESCE(pc.rep_talk_to_listen_ratio,
rm.median_ratio)), 2) AS avg_ratio,
  ROUND(AVG(COALESCE(pc.rep_talk_speed, rm.median_speed)), 2)
AS avg_speed,
```

```
  ROUND(AVG(COALESCE(pc.rep_filler_words, rm.median_filler)),
2) AS avg_filler,
  ROUND(SUM(pc.oppy_value_pitched) * 100.0 /
NULLIF(SUM(pc.oppy_value_served), 0), 2) AS avg_pitch_rate_pct,
  ROUND(SUM(pc.oppy_value_captured) * 100.0 /
NULLIF(SUM(pc.oppy_value_pitched), 0), 2) AS
avg_conversion_rate_pct,
  COUNT(*) AS call_volume
FROM partner_calls pc
JOIN region_medians rm ON pc.account_region = rm.account_region
WHERE pc.oppy_value_served AND pc.oppy_value_pitched > 0 AND
pc.oppy_value_captured > 0
GROUP BY pc.account_region
ORDER BY avg_conversion_rate_pct DESC;
```

Q3 – Accelerator vs TravelAds Pitching

1. Determine revenue performance when Accelerator and TravelAds are pitched together vs. separately.

```
-- Used aggregate functions, filtering records, logical
operators, aggregation, and sorting


SELECT
  COUNT(*) FILTER (
    WHERE EXISTS (
      SELECT 1
      FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%accelerator%'
        AND LOWER(t ->> 'text') LIKE '%travelads%')) AS
both_pitched_calls,

  COUNT(*) FILTER (
    WHERE EXISTS (
      SELECT 1
      FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%accelerator%'
        AND LOWER(t ->> 'text') NOT LIKE '%travelads%')) AS
accelerator_only_calls,

  COUNT(*) FILTER (
    WHERE EXISTS (
      SELECT 1
```

```
      FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%travelads%'
        AND LOWER(t ->> 'text') NOT LIKE '%accelerator%')) AS
travelads_only_calls,

  ROUND(AVG(oppy_value_captured) FILTER (
    WHERE EXISTS (
      SELECT 1 FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%accelerator%'
        AND LOWER(t ->> 'text') LIKE '%travelads%')), 2) AS
avg_revenue_both,

  ROUND(AVG(oppy_value_captured) FILTER (
    WHERE EXISTS (
      SELECT 1 FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%accelerator%'
        AND LOWER(t ->> 'text') NOT LIKE '%travelads%')), 2) AS
avg_revenue_accelerator_only,

  ROUND(AVG(oppy_value_captured) FILTER (
    WHERE EXISTS (
      SELECT 1 FROM jsonb_array_elements(transcript_json) AS t
      WHERE LOWER(t ->> 'text') LIKE '%travelads%'
        AND LOWER(t ->> 'text') NOT LIKE '%accelerator%')), 2)
AS avg_revenue_travelads_only
FROM partner_calls
WHERE oppy_value_captured > 0;
```

2. Using oppy_pitched_column, a different number is identified, giving a hint of difference in recorded text and mention opportunities marked as pitched

```
SELECT

  -- Classify pitch strategy into 4 buckets, CASE statement

  CASE

    WHEN UPPER(oppies_pitched) LIKE '%ACCELERATOR%' AND
UPPER(oppies_pitched) LIKE '%TRAVELADS%' THEN 'Both'

    WHEN UPPER(oppies_pitched) LIKE '%ACCELERATOR%' THEN
'Accelerator only'

    WHEN UPPER(oppies_pitched) LIKE '%TRAVELADS%' THEN
'TravelAds only'
```

```
    ELSE 'None'
  END AS pitch_combo,
  COUNT(*) AS calls,
  ROUND(AVG(oppy_value_captured), 2) AS avg_conversion_value,
  SUM(oppy_value_captured) AS total_revenue
FROM partner_calls
-- Focus only on successful conversion cases
WHERE oppy_value_captured > 0
GROUP BY pitch_combo
ORDER BY avg_conversion_value DESC;
```

Q4 – Segment and Region Performance

Analyze pitch and conversion performance by property type and region.

```
-- Used aggregate functions, filtering records, logical
operators, aggregation, sorting


SELECT
  propertytype__c,
  account_region,
  COUNT(*) AS total_calls,
  ROUND(AVG(oppy_value_pitched), 2) AS avg_pitched,
  ROUND(AVG(oppy_value_captured), 2) AS avg_captured,
  ROUND(SUM(oppy_value_captured), 2) AS total_captured_value,
  ROUND(SUM(oppy_value_pitched) * 100.0 /
NULLIF(SUM(oppy_value_served), 0), 2) AS avg_pitch_rate_pct,
  ROUND(SUM(oppy_value_captured) * 100.0 /
NULLIF(SUM(oppy_value_pitched), 0), 2) AS
avg_conversion_rate_pct
FROM partner_calls
WHERE oppy_value_served > 0
  AND oppy_value_pitched > 0
  AND oppy_value_captured > 0
GROUP BY propertytype__c, account_region
ORDER BY avg_conversion_rate_pct DESC;
```

Q5 – Recommendations Based on Rep Conversion

Identify reps/teams with the highest conversion to find best practices.

```
-- Used aggregate functions, filtering records, aggregation,
sorting


SELECT team,
       ROUND(AVG(rep_talk_to_listen_ratio), 2) AS avg_ratio,
       ROUND(SUM(oppy_value_captured) * 100.0 /
NULLIF(SUM(oppy_value_pitched), 0), 2) AS conversion_pct
FROM partner_calls
WHERE oppy_value_pitched > 0 AND oppy_value_captured > 0

GROUP BY team
ORDER BY conversion_pct DESC;
```

Q6 – Business Impact Simulation

Estimate the impact if all reps performed like the top 25% (Q3 performers).

```
-- Created CTE and its output in the main query, statistics,

-- Used aggregate functions, filtering records


WITH stats AS (
  SELECT
    PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY
rep_talk_to_listen_ratio) AS q3_ratio,
    PERCENTILE_CONT(0.75) WITHIN GROUP (
      ORDER BY oppy_value_captured * 1.0 /
NULLIF(oppy_value_pitched, 0)
    ) AS q3_conversion
  FROM partner_calls
  WHERE oppy_value_pitched > 0
)
SELECT                                              -- Main query
  COUNT(*) FILTER (WHERE rep_talk_to_listen_ratio >=
stats.q3_ratio) AS high_performers,
  ROUND(AVG(oppy_value_captured) FILTER (
    WHERE rep_talk_to_listen_ratio >= stats.q3_ratio
  ), 2) AS avg_high_performer_value,
  ROUND(AVG(oppy_value_captured), 2) AS avg_overall_value
```

```
FROM partner_calls;
```