"Airbnb Data Analysis and Insights: Exploring Listings, Prices, and Reviews"

Trisha Gupta

Que-1) What are the average and maximum prices for each neighbourhood in the "listing" table?

Query:

```
SELECT
  neighbourhood,
  ROUND(AVG(price),2) AS avg_price,
  ROUND(MAX(price),2) AS max_price
FROM `airbnb.listing_airbnb`
GROUP BY neighbourhood
ORDER BY avg_price DESC, max_price DESC
```

Output Result:

Row	neighbourhood	avg_price	max_price
1	Centrum-Oost	308.9	12000.0
2	Centrum-West	303.01	10000.0
3	Zuid	281.21	2000.0
4	De Pijp - Rivierenbuurt	265.91	1250.0
5	Oud-Noord	252.13	2647.0
6	De Baarsjes - Oud-West	245.95	2500.0
7	De Aker - Nieuw Sloten	241.76	2450.0
8	IJburg - Zeeburgereiland	241.16	2575.0
9	Westerpark	234.79	1239.0
10	Watergraafsmeer	234.53	750.0

Que-2) Calculate the rolling average of the number of reviews per month for each listing over a window of 3 months in the "reviewing" table.

Query:

```
SELECT
  listing_id,
  ROUND(AVG(reviews) OVER(PARTITION BY listing_id ORDER BY date_month ROWS BETWEEN 2 PRECEDING AND CURRENT ROW),2) AS rolling_avg
FROM(SELECT
  listing_id,
  COUNT(*) AS reviews,
  EXTRACT(MONTH FROM date) AS date_month
FROM `airbnb.reviews_airbnb`
GROUP BY listing_id, date_month
ORDER BY listing_id ASC, date_month ASC
) AS A
ORDER BY listing_id ASC
```

Output Result:

JOB INFORMATION | RESULTS | JSON | EXECUTION DET

Row	listing_id	date_month	rolling_avg
1	2818	1	14.0
2	2818	2	13.0
3	2818	3	17.0
4	2818	4	24.67
5	2818	5	35.33
6	2818	6	40.67
7	2818	7	38.67
8	2818	8	32.33
9	2818	9	31.33

Que-3) Find the hosts who have all their listings booked for a continuous period of at least 30 days (no gaps)

Query:

```
WITH Ranked_calender AS(SELECT
  l.host_id,
  c.daTRe,
  LAG(c.daTRe) OVER (PARTITION BY l.host_id ORDER BY c.daTRe) AS prev_date
FROM `airbnb.calendar_airbnb` AS c
JOIN `airbnb.listing_airbnb` AS l
ON c.lisTRing_id = l.id
WHERE c.available = "FA"),
CTE AS(
SELECT
  host_id,
  daTRe,
  prev_date,
  DATE_DIFF(daTRe, prev_date, DAY) AS day_diff
FROM Ranked_calender)
```

```
SELECT DISTINCT
  host_id,
  day_diff
FROM CTE
WHERE NOT EXISTS(
  SELECT
    host_id
  FROM CTE AS c
  WHERE CTE.host_id = c.host_id AND day_diff>=30
)
```

Output Result:

Row	host_id	day_diff
1	3159	1
2	3159	null
3	3159	4
4	3159	2
5	3592	null
6	3592	8
7	3592	1
8	42599	null
9	42599	1

Que-4) Identify top 5 neighbourhoods with highest ratio of average price to the number of review in 'listing table'.

Query:

```
SELECT
  neighbourhood,
  ROUND(AVG(price)/COUNT(number_of_reviews),2) AS price_to_review_ratio
FROM `airbnb.listing_airbnb`
GROUP BY neighbourhood
HAVING COUNT(number_of_reviews) > 0
ORDER BY price_to_review_ratio DESC
LIMIT 5
```

Output Result:

Row	neighbourhood	price_to_review_ratio
1	Bijlmer-Oost	5.15
2	Osdorp	4.22
3	De Aker - Nieuw Sloten	3.9
4	Bijlmer-Centrum	3.11
5	Gaasperdam - Driemond	2.62

Q-5) Calculate the total revenue generated by each host for each quarter of the year, considering only the completed bookings in the "calendar" table.

Query:

```
SELECT
  listRing_id,
  EXTRACT(QUARTER FROM daTRe) AS quar_date,
  SUM(price) AS price_sum
FROM `airbnb.calendar_airbnb`
WHERE available = 'FA'
GROUP BY listRing_id, quar_date
ORDER BY listRing_id,quar_date
```

Output Result:

Row	listRing_id	quar_date	price_sum
1	2818	1	6279
2	2818	2	6141
3	2818	3	5934
4	2818	4	6348
5	155548	1	18200
6	155548	2	18000
7	155548	3	18400
8	155548	4	18400
9	162467	2	5020
10	162467	3	15820

Q-6) Find the top 5 hosts with the highest average revenue per month for the last 6 months, considering seasonal variations in price and bookings in the "calendar" table.

Query:

```
WITH RevenueDate AS(SELECT
  FORMAT_DATE('%Y-%m', daTRe) AS years_month,
  SUM(price) AS revenue_sum,
  listRing_id
FROM `airbnb.calendar_airbnb`
WHERE available = "FA" AND daTRe >= DATE_SUB(CURRENT_DATE, INTERVAL 6 MONTH)
GROUP BY listRing_id, years_month),

avgrev AS(SELECT
  ROUND(SUM(revenue_sum)/COUNT(DISTINCT years_month),2) AS avg_revenue_per_month,
  listRing_id
FROM RevenueDate
GROUP BY listRing_id)

SELECT DISTINCT
  listRing_id,
  avgrev.avg_revenue_per_month
FROM avgrev
ORDER BY avgrev.avg_revenue_per_month DESC
LIMIT 5
```

Output Result:

Row	lisTRing_id	avg_revenue_per_mo
1	39431135	221807.69
2	2474754	173333.33
3	38889961	61600.0
4	20840176	55452.75
5	19528180	48333.33

Que-7) Calculate the percentage change in the number of bookings for each neighbourhood compared to the previous year, considering seasonal trends.

Query:

```
WITH CTE1 AS(SELECT
  l.neighbourhood,
  c.daTRe,
  FORMAT_DATE('%Y-%m', c.daTRe) AS years_month,
  COUNT(*) AS no_booking
FROM `airbnb.listing_airbnb` AS l
JOIN `airbnb.calendar_airbnb` AS c
ON l.id = c.lisTRing_id
WHERE c.available = "FA"
GROUP BY l.neighbourhood, years_month, c.daTRe
ORDER BY l.neighbourhood ASC
),
CTE2 AS (SELECT
  EXTRACT(YEAR FROM DATE_SUB(CURRENT_DATE,INTERVAL 1 YEAR)) AS prev_year,
  neighbourhood,
  SUM(no_booking) AS prev_year_no_booking
FROM CTE1
WHERE years_month >= FORMAT_DATE('%Y-%m', DATE_SUB(CURRENT_DATE, INTERVAL 1 YEAR))
GROUP BY neighbourhood, prev_year)

SELECT
  CTE1.years_month,
  CTE1.neighbourhood,
  CTE1.no_booking AS current_booking,
  CTE2.prev_year_no_booking AS prev_booking,
  ROUND(((CTE1.no_booking - CTE2.prev_year_no_booking)/ CTE2.prev_year_no_booking) *100, 2) AS precent_change
FROM CTE1
JOIN CTE2
ON CTE1.neighbourhood = CTE2.neighbourhood
-- AND CTE1.years_month = DATE_ADD(prev_year, INTERVAL 1 YEAR)
ORDER BY CTE1.years_month,CTE1.neighbourhood
```

Output Result:

Row	years_month	neighbourhood	current_booking	prev_booking	precent_change
1	2023-06	Bijlmer-Centrum	7	6126	-99.89
2	2023-06	Bijlmer-Centrum	20	6126	-99.67
3	2023-06	Bijlmer-Centrum	19	6126	-99.69
4	2023-06	Bijlmer-Centrum	18	6126	-99.71
5	2023-06	Bijlmer-Centrum	18	6126	-99.71
6	2023-06	Bijlmer-Centrum	17	6126	-99.72
7	2023-06	Bijlmer-Centrum	16	6126	-99.74
8	2023-06	Bijlmer-Centrum	15	6126	-99.76
9	2023-06	Bijlmer-Centrum	17	6126	-99.72

Que-8) Find the top 10 busiest hosts based on the number of total bookings, considering fluctuations in the number of listings they manage over time.

Query:

```
WITH CTE1 AS(SELECT
  c.lisTRing_id,
  l.host_id,
  COUNT(*) AS no_booking
FROM `airbnb.listing_airbnb` AS l
JOIN `airbnb.calendar_airbnb` AS c
ON l.id = c.lisTRing_id
WHERE c.available = "FA"
GROUP BY c.lisTRing_id, l.host_id),

CTE2 AS(SELECT
  host_id,
  COUNT(DISTINCT lisTRing_id) AS total_listing
FROM CTE1
GROUP BY host_id),

CTE3 AS(SELECT
  host_id,
  SUM(no_booking) AS total_booking
FROM CTE1
GROUP BY host_id)

SELECT
  CTE2.host_id,
  CTE2.total_listing,
  CTE3.total_booking,
  CTE2.total_listing/ CTE3.total_booking AS avg_boooking_per_listing
FROM CTE2
JOIN CTE3 ON CTE2.host_id = CTE3.host_id
ORDER BY CTE3.total_booking DESC
```

Output Result:

Row	host_id	total_listing	total_booking	avg_boooking_per_li
1	203731852	18	3659	0.004919376878...
2	9282300	8	2489	0.003214142225...
3	237150996	4	1460	0.002739726027...
4	2028878	4	1420	0.002816901408...
5	198405490	4	1417	0.002822865208...
6	46691672	4	1388	0.002881844380...
7	273266448	4	1338	0.002989536621...
8	219618697	6	1216	0.004934210526...
9	503491020	6	1199	0.005004170141...

Que-9) Identify the listings that have the highest average rating for each neighborhood, considering both "listing" and "reviews" tables.

Query:

```
WITH CTE1 AS(
SELECT
  r.listing_id,
  AVG(l.number_of_reviews) AS no_review
FROM `airbnb.reviews_airbnb` AS r
JOIN `airbnb.listing_airbnb` AS l
ON r.listing_id = l.id
GROUP BY r.listing_id),
CTE2 AS(SELECT
  l.neighbourhood,
  l.id,
  no_review,
  DENSE_RANK() OVER (PARTITION BY l.neighbourhood  ORDER BY no_review DESC) AS rn
FROM `airbnb.listing_airbnb` AS l
JOIN CTE1 ON  l.id = CTE1.listing_id
)
```

```sql
SELECT
  CTE2.neighbourhood,
  CTE2.id,
  CTE2.no_review
FROM CTE2
JOIN `airbnb.listing_airbnb` AS l
ON CTE2.id = l.id
WHERE rn = 1
```

Output Result :

Row	neighbourhood	id	no_review
1	Bijlmer-Centrum	1149380	556.0
2	De Baarsjes - Oud-West	32485135	1934.0
3	Gaasperdam - Driemond	4459764	406.0
4	De Aker - Nieuw Sloten	884329	518.0
5	Bijlmer-Oost	2942229	338.0
6	Noord-West	4282686	401.0
7	Oud-Oost	35927687	1334.0
8	Centrum-West	82482	1005.0
9	Watergraafsmeer	4147932	339.0

Que-10) Calculate the cumulative revenue for each host over time, considering both "calendar" and "listing" tables.

Query:

```sql
WITH revenuedata AS(SELECT
  l.host_id,
  c.daTRe,
  l.price,
  ROW_NUMBER() OVER (PARTITION BY l.host_id ORDER BY c.daTRe) AS rn
FROM `airbnb.calendar_airbnb` AS c
JOIN `airbnb.listing_airbnb` AS l
ON c.lisTRing_id = l.id
ORDER BY l.host_id)
SELECT
  host_id,
  daTRe,
  SUM(price) OVER ( PARTITION BY host_id ORDER BY daTRe, rn) AS cumulative_revenue
FROM revenuedata
ORDER BY host_id
```

Output Result:

Row	host_id	daTRe	cumulative_revenue
1	3159	2023-06-05	69
2	3159	2023-06-06	138
3	3159	2023-06-07	207
4	3159	2023-06-08	276
5	3159	2023-06-09	345
6	3159	2023-06-10	414
7	3159	2023-06-11	483
8	3159	2023-06-12	552
9	3159	2023-06-13	621
10	3159	2023-06-14	690

Que-11) Identify the neighborhoods with the highest seasonal price fluctuations (difference in average price between the peak and off-peak seasons).

Query:

```
WITH CTE1 AS(SELECT
  l.neighbourhood,
  EXTRACT(MONTH FROM c.daTRe) AS mont,
  ROUND(AVG(c.price),2) AS avg_price
FROM `airbnb.calendar_airbnb` AS c
JOIN `airbnb.listing_airbnb` AS l
ON c.lisTRing_id = l.id
WHERE c.available = 'FA'
GROUP BY mont, l.neighbourhood
ORDER BY l.neighbourhood ASC, mont ASC),
CTE2 AS (
SELECT
  neighbourhood,
  MAX(avg_price) AS peak_season
FROM CTE1
GROUP BY neighbourhood),
```

```
98  CTE3 AS (
99  SELECT
100   neighbourhood,
101   MIN(avg_price) AS off_peak
102 FROM CTE1
103 GROUP BY neighbourhood),
104 CTE4 AS (
105 SELECT
106   CTE1.neighbourhood,
107   CTE2.peak_season,
108   CTE3.off_peak,
109   CTE2.peak_season - CTE3.off_peak AS price_fluctuation
110 FROM CTE2
111 JOIN CTE3 ON CTE3.neighbourhood = CTE2.neighbourhood
112 JOIN CTE1 ON CTE2.neighbourhood = CTE1.neighbourhood)
113 SELECT
114   neighbourhood,
115   ROUND(MAX(CTE4.price_fluctuation),2) AS highest_price_fluc
116 FROM CTE4
117 GROUP BY neighbourhood
118 ORDER BY highest_price_fluc DESC
```

Output Result:

Row	neighbourhood	highest_price_fluc
1	Zuid	58.91
2	Osdorp	48.45
3	De Aker - Nieuw Sloten	40.01
4	IJburg - Zeeburgereiland	37.78
5	Oud-Noord	34.77
6	Centrum-West	27.15
7	Bijlmer-Centrum	26.21
8	De Pijp - Rivierenbuurt	23.89
9	Noord-Oost	22.77
10	Centrum-Oost	21.04

Insight:

1- Average and Maximum Prices by Neighbourhoods:

Insight: You can identify which neighbourhoods have the highest and lowest average and maximum prices for listings. This can help hosts understand price trends in different areas and adjust their pricing strategy accordingly.

Action: Hosts can set competitive prices based on the average and maximum prices in their neighbourhoods. They can also analyse why certain neighbourhoods have higher prices and whether it's due to amenities, location, or other factors.

2- Rolling Average of Reviews per Month:

Insight: The rolling average of reviews per month can show trends in guest satisfaction over a 3-month period. It helps to understand how the quality of listings is perceived by guests over time.

Action: Hosts can monitor their listings' performance and make improvements if the rolling average of reviews per month is declining. Consistently positive reviews are crucial for attracting new guests.

3- Continuous Bookings for Hosts:

Insight: Identifying hosts with all their listings booked for at least 30 days without gaps indicates high demand and occupancy. It highlights efficient management and popular listings.

Action: Hosts can leverage this insight to optimize pricing, extend similar offerings, or provide incentives to guests for longer stays.

4- Neighbourhoods with High Price-to-Review Ratio:

Insight: The ratio of average price to the number of reviews highlights the value perceived by guests for different neighbourhoods. Higher ratios may indicate better value.

Action: Hosts can adjust their pricing and improve amenities or services to align with guest expectations, especially in neighbourhoods with lower price-to-review ratios.

5- Revenue by Host and Quarter:

Insight: Calculating revenue by host for each quarter identifies seasonal revenue trends. It helps understand peak booking periods.

Action: Hosts can plan their marketing efforts and resource allocation based on seasonal revenue patterns. They can offer discounts during off-peak seasons to boost bookings.

6- Hosts with High Average Revenue per Month:

Insight: Identifying hosts with high average revenue per month considering seasonal variations suggests effective pricing and occupancy strategies.

Action: Hosts can learn from these high-performing hosts, adjust their pricing strategy, and optimize their listings for higher revenue.

7- Percentage Change in Bookings by Neighbourhoods:

Insight: Monitoring the percentage change in bookings by neighbourhoods helps understand how demand is evolving.

Action: Hosts can tailor their marketing efforts, adjust pricing, and offer promotions during periods of decreased demand.

8- Busiest Hosts with Fluctuating Listings:

Insight: Identifying busiest hosts considering fluctuations in listings shows how well hosts manage changing inventory.

Action: Hosts can learn from these top hosts and optimize their management strategies to ensure smooth operations even with changing listing availability.

9- Average Rating by Neighbourhoods:

Insight: Identifying listings with the highest average rating by neighbourhoods highlights hosts offering exceptional guest experiences.

Action: Hosts can learn from these listings, improve their own guest experiences, and address areas for improvement based on neighbourhood-specific expectations.

10- Cumulative Revenue Over Time:

Insight: Calculating cumulative revenue for each host over time reflects long-term earnings trends.

Action: Hosts can gauge the sustainability of their hosting business, analyze trends in revenue growth, and make informed decisions.

11- Seasonal Price Fluctuations:

Insight: Identifying neighbourhoods with the highest seasonal price fluctuations helps understand how pricing changes in response to demand shifts.

Action: Hosts can adjust pricing strategies, consider offering discounts during off-peak seasons, and optimize revenue during peak seasons.