SQL Query Evaluation: Optimizing Solutions for Identifying Start and End Locations per Customer

Introduction:

In this document, we evaluate different SQL solutions for a common Data Structures and Algorithms (DSA) problem: identifying the starting and ending locations for each customer based on travel data. The goal is to assess the optimization, problem-solving approach, and efficiency of each solution to determine the best approach.

Problem Statement:

Given a dataset containing customer travel data with start and end locations, the task is to write SQL queries that identify the starting and ending locations for each customer. There's no fixed order for which location is considered the start or end, making this a challenge in data manipulation and query design.

Solution Overview:

Solution 1:

This solution uses a Common Table Expression (CTE) to first gather all unique start and end locations for each customer. It then joins this CTE with the original travel data table to identify which location corresponds to the start and which to the end for each customer. The approach ensures that only unique start or end locations are considered per customer, managing the ambiguity of start and end points effectively.

Solution 2:

In this approach, SQL window functions are leveraged to partition data by customer and location. By counting occurrences of each location within these partitions, the query identifies which locations occur exactly once per customer, thus determining the start and end locations. This method simplifies the aggregation process and optimizes performance by reducing the need for multiple joins or conditional checks.

Solution 3:

This solution utilizes multiple left joins on the travel data table to directly match start and end locations. It checks for matches where the start location of one record matches the end location of another record for the same customer. Although straightforward in concept, this approach may involve more complex join operations and could potentially be less efficient with larger datasets compared to the other solutions.

Solutions:

Solution 1 – My Solution

```
With CTE as (
Select customer, locations, count(locations) as points from (
Select c1.customer, c1.start_loc as locations from travel_data c1
union all
Select c2.customer, c2.end_loc from travel_data c2
)A
group by customer, locations
having count(locations) = 1)
/*---with this cte, we'll identify the points where either it started or ended not sure which is one,
but for sure i got it that both start and end will have a count of 1 */
/* so in next step, i'll join with travel data, on customer and where locations from cte matches with either the start
or end loc */
Select cte.customer,
max(case when cte.locations = t.start_loc then locations end) as Starting_Location,
max(case when cte.locations = t.end_loc then locations end) as end_Location
from cte
inner join travel_data t
on cte.customer = t.customer and (cte.locations = t.start_loc or cte.locations = t.end_loc)
group by cte.customer
order by cte.customer;
```

Analysis for Solution 1:

Criteria	Solution 1
Optimization and Efficiency	**Moderate:** Uses a CTE to filter unique locations, but may involve redundant checks.
Problem-Solving Approach	**Good:** Successfully identifies single occurrences using a CTE and conditional aggregation.
Smartness	**Moderate:** Effective use of CTE, though potential for simplification.
Energy Efficiency	**Moderate:** Additional joins and checks may impact efficiency.

Solution 2 – Ankit’s Solution – Aam Jindagi

```
with cte as (
Select customer, locations, count(locations) over(partition by customer, locations) as points, column_name from (
Select c1.customer, c1.start_loc as locations, 'start_loc' as column_name from travel_data c1
union all
Select c2.customer, c2.end_loc, 'end_loc' as column_name from travel_data c2
)A)
Select customer
,max(case when column_name = 'start_loc' then locations end) as starting_location
,max(case when column_name = 'end_loc' then locations end) as end_location
from cte
where points = 1
group by customer;
```

Analysis for Solution 2:

Criteria	Solution 2
Optimization and Efficiency	**High:** Utilizes SQL window functions to efficiently count occurrences, optimizing performance.
Problem-Solving Approach	**Good:** Effectively partitions and aggregates data using window functions.
Smartness	**High:** Smart use of advanced SQL features (window functions).
Energy Efficiency	**High:** Avoids unnecessary joins, optimizing resource usage.

Solution 3: Mentos Zindagi (my favorite):

```
Select t1.customer
,max(case when t2.start_loc is null then t1.start_loc end) as starting_location
,max(case when t3.start_loc is null then t1.end_loc end ) as end_location
from travel_data t1
left join travel_data t2 on t1.customer = t1.customer and t1.start_loc = t2.end_loc
left join travel_data t3 on t1.customer = t3.customer and t1.end_loc = t3.start_loc
group by t1.customer;
```

Analysis for Solution 3:

Criteria	Solution 3
Optimization and Efficiency	**Low:** Relies on multiple left joins, potentially less efficient with larger datasets.
Problem-Solving Approach	**Fair:** Matches start and end locations directly using joins.
Smartness	**Low:** Traditional join operations without leveraging advanced SQL features.
Energy Efficiency	**Low:** Performs more join operations than necessary, impacting efficiency.

Summary and Evaluation of SQL Solutions for Identifying Start and End Locations per Customer

Solution	Optimization and Efficiency	Problem-Solving Approach	Smartness	Energy Efficiency
Solution 1	Moderate	Good	Moderate	Moderate
Solution 2	High	Good	High	High
Solution 3	Low	Fair	Low	Low

Final Analysis:

- Solution 2 emerges as the winner due to its high scores across all evaluation criteria:
  - *Optimization and Efficiency:* High - Uses SQL window functions effectively to streamline the query.
  - *Problem-Solving Approach:* Good - Clears partitions and aggregates data accurately.
  - *Smartness:* High - Makes efficient use of advanced SQL features.
  - *Energy Efficiency:* High - Optimizes resource usage by avoiding unnecessary joins.

Why Solution 2 Wins:

- Solution 2 demonstrates the best balance of efficiency, clarity, and optimal SQL feature usage. It effectively solves the problem of identifying start and end locations per customer with high performance and minimal complexity, making it the recommended approach for this task.