Rapport - Visites dans une trajectoire

1. **Step 1: Extracting the Previous POI Information (`trajectoire1`)**:

```
select userId, poi, datetime, rank, interval_duration, traj,
     COALESCE(LAG(poi) OVER (PARTITION BY userId, traj ORDER BY rank), poi) AS prev_poi
from visit_traj2
order by userId, traj, rank
```

   - This step extracts the information from the `visit_traj2` table and creates a new column called `prev_poi` that holds the value of the previous point of interest (POI) for each row. The `LAG()` function is used to retrieve the value of the POI in the previous row, partitioned by `userId` and `traj` (trajectory) and ordered by the `rank`.
   - If the value is `NULL`, the current POI value is used (`COALESCE` is used for this).
   - This helps in tracking the previous POI, which is essential for detecting consecutive visits.

2. **Step 2: Assigning a Rank to Each Row within Each Trajectory (`trajectoire2`)**:

```
select *, ROW_NUMBER() OVER (PARTITION BY userId, traj ORDER BY rank) AS rank_in_traj
from trajectoire1
order by userId, traj, rank_in_traj
```

   - This step adds a new column called `rank_in_traj`, which assigns a unique rank to each row within each trajectory. The rows are ordered by the `rank` value within each partition of `userId` and `traj`.
   - The purpose of `rank_in_traj` is to uniquely identify each record within a trajectory to help track their sequence.

3. **Step 3: Identifying New POI Visit Segments (`trajectoire3`)**:

```
select *, CASE
    WHEN rank_in_traj = 1 THEN 1
    WHEN prev_poi IS NULL OR prev_poi != poi THEN 1
    ELSE 0
  END AS debut_poi
from trajectoire2
order by userId, traj, rank
```

   - A new column called `debut_poi` is added to identify the start of a new POI visit segment.
   - The logic is as follows:
     - If the row is the first in the trajectory (`rank_in_traj = 1`), it is considered the start of a new visit (`debut_poi = 1`).
     - If the previous POI (`prev_poi`) is different from the current POI, it indicates the start of a new POI visit (`debut_poi = 1`).
     - Otherwise, it is a continuation of the current POI visit (`debut_poi = 0`).

4. **Step 4: Creating a Unique Visit ID for Each POI Visit (`trajectoire4`)**:

```
select *,
    sum(debut_poi) over (partition by userid, traj order by rank rows between unbounded preceding and current row) as visitid
from trajectoire3
order by userId, traj, rank
```

   - This step assigns a unique `visitid` to each POI visit by calculating the cumulative sum of `debut_poi` within each trajectory (`userId`, `traj`).
   - The cumulative sum ensures that each new POI visit gets a unique identifier while consecutive visits to the same POI have the same `visitid`.

5. **Step 5: Aggregating Visits to the Same POI (`trajectoire5`)**:

```
select userid, traj, visitid, poi,
    min(datetime) as date_in,
    max(datetime) as date_out,
    count(*) as nb_checkin
from trajectoire4
group by userid, traj, visitid, poi
order by userid, traj, visitid
```

   - This step aggregates the visits for each POI based on the `visitid`.
   - The `min(datetime)` is used to get the start time (`date_in`), and `max(datetime)` is used to get the end time (`date_out`) of the visit.
   - The `count(*)` gives the number of check-ins (`nb_checkin`) during the visit.
   - This creates a record for each aggregated POI visit, including the start and end times.

6. **Step 6: Calculating the Duration for Each POI Visit (`trajectoire6`)**:

```
SELECT T1.userId, T1.traj, T1.visitId, T2.poi, T1.nb_checkin, T1.date_in, T1.date_out,
    CAST(EXTRACT(EPOCH FROM (T1.date_out - T1.date_in)) AS INTEGER) AS duration
FROM trajectoire5 T1 JOIN trajectoire4 T2
ON T1.userId = T2.userId AND T1.traj = T2.traj AND T1.visitId = T2.visitId
GROUP BY T1.userId, T1.traj, T1.visitId, T2.poi, T1.nb_checkin, T1.date_in, T1.date_out
ORDER BY T1.userId, T1.traj, T1.visitId
```

   - In this step, the duration of each POI visit is calculated by taking the difference between `date_out` and `date_in` and extracting the duration in seconds (`EXTRACT(EPOCH FROM ...)`). The duration is then cast to an integer.
   - This query joins `trajectoire5` and `trajectoire4` to enrich the visit information.