Analysis of Patient Admissions: Initial Admissions, Readmissions, and Admission Status Overview

Creating temp table for encounters

```
CREATE TEMP TABLE temp_encounters AS
SELECT * FROM encounters;
```

Adding a column to check for rows that need to be dropped (there are overlap between admissions)

```
ALTER TABLE temp_encounters ADD COLUMN drop_row BOOLEAN;
```

Some admission records overlap with previous ones, so we identify and remove those duplicate entries

```
DO $$
DECLARE
      e_id UUID; -- to hold encounter id
    p_id UUID; --  to hold patient id
    drop_row_check BOOLEAN; -- check to drop a row or not
    prev_discharge_date TIMESTAMP; -- to hold prev stop time
    order_column_1 TIMESTAMP; -- to hold start time
    order_column_2 TIMESTAMP; -- to hold stop time
    last_patient UUID := NULL;

    cur CURSOR FOR
        SELECT id, patient, start, stop
        FROM temp_encounters
        ORDER BY patient, start ASC, stop ASC;
BEGIN
    OPEN cur;
    LOOP
        FETCH cur INTO e_id, p_id, order_column_1, order_column_2;
        EXIT WHEN NOT FOUND;

           IF last_patient IS DISTINCT FROM p_id THEN
            prev_discharge_date := NULL;  -- Reset prev_discharge_date it means now we
checking for new patient
        END IF;
           IF order_column_1 < prev_discharge_date THEN
            drop_row_check = TRUE;
                prev_discharge_date = GREATEST(order_column_2, prev_discharge_date);
        ELSE
                prev_discharge_date = order_column_2;
                drop_row_check = FALSE;
        END IF;

        UPDATE temp_encounters
        SET drop_row = drop_row_check
        WHERE patient = p_id AND start = order_column_1 AND stop = order_column_2;

        last_patient := p_id;

    END LOOP;
    CLOSE cur;
END $$;
```

Removing records that overlaps

```
DELETE FROM temp_encounters
WHERE drop_row = TRUE
```

Creating another temp table just to make things clear

```
CREATE TEMP TABLE  admission_analysis AS
 SELECT id,
           patient,
           start,
           stop,
           EXTRACT(DAY FROM (stop - start)) AS admitted_days,
             LAG(stop) OVER (PARTITION BY patient ORDER BY start
ASC, stop ASC) AS prev_discharge,
           EXTRACT(DAY FROM (start - LAG(stop) OVER (PARTITION BY
patient ORDER BY start ASC, stop ASC))) AS
days_interval_between_admission,
             0 AS running_total
    FROM temp_encounters;
```

Adding running total that helps us in identifying readmissions

```
DO $$
DECLARE
    r_t INT := 0; -- variable to update running total
    r_t_c INT := 0; -- variable to hold current running total in each iteration
    admitt_days INT := 0;
    days_interval INT := 0;
    order_column_1 TIMESTAMP; -- to store start time
     order_column_2 TIMESTAMP; --  to store stop time
    p_id UUID; -- to store patient id
    p_d TIMESTAMP; -- to store previous discharge of patient
    cur CURSOR FOR
        SELECT admitted_days, days_interval_between_admission, patient, prev_discharge,
start, stop
        FROM admission_analysis
        ORDER BY patient, start ASC, stop ASC;
BEGIN
    OPEN cur;
    LOOP
        FETCH cur INTO admitt_days, days_interval, p_id, p_d, order_column_1,
order_column_2;
        EXIT WHEN NOT FOUND;

          IF p_d IS NULL THEN
           r_t := 0;
               r_t_c := 0;
        ELSIF admitt_days >= 1 THEN
            r_t := r_t_c + days_interval;
            r_t_c := 0;
        ELSE
            r_t_c := r_t_c + days_interval;
            r_t := r_t_c;
        END IF;

        UPDATE admission_analysis
        SET running_total = r_t
        WHERE patient = p_id AND start = order_column_1 AND stop = order_column_2;
    END LOOP;
    CLOSE cur;
END $$;
```

Adding a column that checks for admission status

```
ALTER TABLE admission_analysis ADD COLUMN admission_status VARCHAR(20);
```

Updating the admission status column with the initial admission, readmission and visited status

```
WITH MinStart AS (
    SELECT
          id,
        patient,
        start,
        admitted_days,
          running_total,
          days_interval_between_admission,
        MIN(CASE WHEN admitted_days >= 1 THEN start END) OVER (PARTITION BY patient)
AS min_start
    FROM admission_analysis
)
UPDATE admission_analysis
SET admission_status = subquery.admission_status
FROM (
    SELECT
          id,
        patient,
        start,
          admitted_days,
        CASE
            WHEN (start = min_start AND admitted_days >= 1) OR (start > min_start AND
running_total > 30 AND admitted_days >= 1 ) THEN 'initial admission'
              WHEN (admitted_days >= 1 AND days_interval_between_admission = 0 AND
running_total = 0) THEN 'continuous admission'
              WHEN (admitted_days >= 1) THEN 'readmission'
              ELSE 'visited'
        END AS admission_status
    FROM MinStart
) AS subquery
WHERE admission_analysis.id = subquery.id AND admission_analysis.patient =
subquery.patient AND admission_analysis.start = subquery.start;
```

Reviewing the table to confirm that all previous queries executed as expected.

```
SELECT * FROM admission_analysis
ORDER BY patient, start ASC, stop ASC;
```

Counting the total number of initial admission

```
SELECT COUNT(*) AS total_initial_admissions
FROM admission_analysis
WHERE admission_status = 'initial admission'
```

Counting the total number of readmissions

```
SELECT COUNT(*) AS total_readmissions
FROM admission_analysis
WHERE admission_status = 'readmission'
```

Counting the total number of admissions

```
SELECT COUNT(*) AS total_admissions
FROM admission_analysis
WHERE admission_status = 'initial admission' OR admission_status = 'readmission'
```

Counting the total number of patients admitted

```
WITH admitted_patients AS (
    SELECT
        DISTINCT patient
    FROM
        admission_analysis
    WHERE
        admission_status IN ('initial admission', 'readmission', 'continuous admission')
)
SELECT
    COUNT(*) AS admitted_patients_count
FROM
    admitted_patients;
```

Counting the total number of patients who have never been admitted

```
WITH admitted_patients AS (
    SELECT
        DISTINCT patient
    FROM
        admission_analysis
    WHERE
        admission_status IN ('initial admission', 'readmission', 'continuous admission')
)
SELECT
    COUNT(DISTINCT patient) AS never_admitted_patients_count
FROM
    admission_analysis
WHERE
    patient NOT IN (SELECT patient FROM admitted_patients);
```

Counting the total number of patients who have been re-admitted

```sql
WITH admitted_patients AS (
    SELECT
        DISTINCT patient
    FROM
        admission_analysis
    WHERE
        admission_status = 'readmission'
)
SELECT
    COUNT(DISTINCT patient) AS readmitted_patients
FROM
    admission_analysis
WHERE
    patient IN (SELECT patient FROM admitted_patients);
```

The Results are:

Total Number of Patients: 974

Total Number of Admissions: 623

Total Number of Initial Admissions: 347

Total Number of Re-admissions: 276

Total Number of Patients Admitted: 143

Total Number of Patients Who Have Never Been Admitted: 831

Total Number of Patients Who Have Re-admitted: 20