CREATE OR REPLACE FUNCTION db\_23\_0853\_ancestry\_2023.sp\_check\_n\_consecutive\_consonant(

in\_batch\_name character varying,

in\_field\_name character varying,

in\_n int)

RETURNS refcursor

LANGUAGE plpgsql

SET search\_path TO 'db\_23\_0853\_ancestry\_2023'

AS $function$

DECLARE

var\_condition varchar;

var\_sql varchar;

cursor\_result refcursor := 'cursor\_result';

BEGIN

/\*

in\_batch\_name: tên batch, empty mean all

in\_field\_name: tên field

in\_n: số lượng phụ âm cần lọc

\*/

var\_regex varchar := '[b-df-hj-np-tv-z]{' || in\_n || '}';

var\_sql := '

SELECT data\_value as ' || quote\_ident(in\_field\_name) || ', amount

FROM sp\_split\_single\_word\_by\_field($1,$2)

WHERE data\_value ~\* ''' || var\_regex || '''

ORDER BY 2 DESC, 1

';

OPEN cursor\_result FOR

EXECUTE

(var\_sql) USING in\_batch\_name,

in\_field\_name;

RETURN cursor\_result;

END;

$function$

Giải thích function (1)

Đây là một đoạn code PL/pgSQL để tạo một hàm có tên db\_23\_0853\_ancestry\_2023.sp\_check\_n\_consecutive\_consonant. Hàm này có chức năng lọc các từ trong một trường dữ liệu cụ thể, trong đó xuất hiện liên tiếp có n phụ âm và trả về dữ liệu qua một con trỏ tham chiếu.

**Giải thích từng dòng lệnh trong hàm như sau:**

1. CREATE OR REPLACE FUNCTION db\_23\_0853\_ancestry\_2023.sp\_check\_n\_consecutive\_consonant(in\_batch\_name character varying, in\_field\_name character varying, in\_n int) RETURNS refcursor LANGUAGE plpgsql SET search\_path TO 'db\_23\_0853\_ancestry\_2023' AS $function$ - Khai báo hàm tạo ra một hàm PL/pgSQL. Các tham số đầu vào của hàm bao gồm tên của batch, tên của trường và số lượng phụ âm cần lọc. Hàm trả về một con trỏ tham chiếu và được định nghĩa để sử dụng trong schema db\_23\_0853\_ancestry\_2023.

2. DECLARE var\_condition varchar; var\_sql varchar; cursor\_result refcursor := 'cursor\_result'; - Định nghĩa các biến cục bộ cho hàm, bao gồm một chuỗi điều kiện và một chuỗi SQL để thực thi. Cuối cùng, một con trỏ tham chiếu cursor\_result được khởi tạo.

3. var\_regex varchar := '[b-df-hj-np-tv-z]{' || in\_n || '}'; - Định nghĩa một biểu thức chính quy được sử dụng để lọc các từ trong trường dữ liệu. Biểu thức này chỉ liên quan đến những phụ âm xuất hiện liên tiếp với số lượng phụ âm được chỉ định bởi in\_n.

4. var\_sql := 'SELECT data\_value as ' || quote\_ident(in\_field\_name) || ', amount FROM sp\_split\_single\_word\_by\_field($1,$2) WHERE data\_value ~\* ''' || var\_regex || ''' ORDER BY 2 DESC, 1'; - Định nghĩa một câu lệnh SQL được sử dụng để truy vấn cơ sở dữ liệu. Câu lệnh này sử dụng hàm sp\_split\_single\_word\_by\_field để phân tách các từ trong trường dữ liệu và lọc ra các từ có liên tiếp số lượng phụ âm được chỉ định bởi in\_n. Câu lệnh trả về hai cột: data\_value - giá trị từ đã lọc và amount - số lần xuất hiện của từ trong trường dữ liệu.

5. OPEN cursor\_result FOR EXECUTE(var\_sql) USING in\_batch\_name, in\_field\_name; - Mở con trỏ tham chiếu cursor\_result và thực thi câu lệnh SQL được định nghĩa trước đó. Các tham số in\_batch\_name và in\_field\_name được sử dụng trong câu lệnh SQL thông qua cấu trúc USING.

6. RETURN cursor\_result; - Trả về con trỏ tham chiếu cursor\_result.

Giải thích function (2)

Đây là một function trong PostgreSQL để tìm kiếm các từ có số lượng phụ âm liên tiếp lớn hơn hoặc bằng giá trị được xác định.

**Các tham số đầu vào:**

- in\_batch\_name: Tên batch, nếu để trống có nghĩa là tìm trong tất cả các batch.

- in\_field\_name: Tên của trường chứa dữ liệu cần tìm kiếm.

- in\_n: Số lượng phụ âm liên tiếp cần tìm kiếm.

**Các hàm được sử dụng:**

1. SET search\_path TO 'db\_23\_0853\_ancestry\_2023': Thiết lập schema mặc định cho function.

2. quote\_ident(): Hàm này nhận vào một chuỗi và trả về chuỗi đã được quote để sử dụng như một identifier hợp lệ trong cú pháp SQL. Trong function này, nó được sử dụng để quote tên trường (in\_field\_name).

3. sp\_split\_single\_word\_by\_field(): Đây là một stored procedure (lưu trữ) khác trong cùng schema với function hiện tại. Nó được sử dụng để phân tách các từ trong trường dữ liệu thành các record riêng biệt, mỗi record chứa một từ. Stored procedure này nhận hai tham số đầu vào: tên batch và tên trường, và trả về một bảng với các cột sau:

- batch\_name: Tên batch.

- field\_name: Tên trường.

- data\_value: Giá trị của từng từ đã được phân tách.

- row\_number: Số thứ tự của record tương ứng.

4. OPEN cursor\_result FOR EXECUTE(var\_sql) USING in\_batch\_name, in\_field\_name;: Hàm này dùng để mở một cursor và lấy dữ liệu từ câu lệnh SQL. Câu lệnh SQL được lưu trong biến var\_sql, và giá trị của in\_batch\_name và in\_field\_name được truyền vào sử dụng USING.

**Giải thích:**

Trong function này, trước khi thực hiện câu lệnh SQL để tìm kiếm các từ có số lượng phụ âm liên tiếp lớn hơn hoặc bằng giá trị được xác định (in\_n), ta thiết lập schema mặc định cho function và sử dụng hàm quote\_ident() để quote tên trường (in\_field\_name). Sau đó, ta sử dụng stored procedure sp\_split\_single\_word\_by\_field() để phân tách các từ trong trường dữ liệu thành các record riêng biệt. Kết quả trả về từ stored procedure này được sử dụng để thực hiện câu lệnh SQL để tìm kiếm các từ có số lượng phụ âm liên tiếp lớn hơn hoặc bằng giá trị được xác định (in\_n). Cuối cùng, ta sử dụng hàm OPEN cursor\_result FOR EXECUTE(var\_sql) USING in\_batch\_name, in\_field\_name; để mở một cursor và lấy dữ liệu từ câu lệnh SQL. Kết quả cuối cùng của function là một cursor.