CREATE

OR

REPLACE

FUNCTION lookup\_ancestry.master\_lookup(json) RETURNS

TABLE

(single\_words json, multi\_words json) LANGUAGE plpgsql AS $function $DECLARE json\_parameters alias for $1;

result\_single json;

result\_multi json;

var\_sql varchar;

value\_single varchar;

value\_multi varchar;

value\_set varchar;

value\_country varchar;

value\_locale varchar;

value\_field\_name varchar;

value\_field\_name\_cond varchar;

sqlLK varchar;

slqConditionCountry varchar;

v\_f json;

v\_s json;

value\_table\_name varchar;

rcResult refcursor;

BEGIN

set

search\_path = lookup\_ancestry;

result\_single := '{"ERROR":"NOT SUPPORT"}' ::json;

result\_multi := '{}' ::json;

value\_table\_name := '';

value\_single := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'main\_word') ::text, '"', '')

)

),

''

);

value\_multi := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'full\_word') ::text, '"', '')

)

),

''

);

value\_table\_name := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'value\_set') ::text, '"', '')

)

),

''

);

value\_country := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'language') ::text, '"', '')

)

),

''

);

value\_locale := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'locale') ::text, '"', '')

)

),

''

);

value\_field\_name := COALESCE(

TRIM(

LOWER(

replace

((json\_parameters -> 'field\_name') ::text, '"', '')

)

),

''

);

value\_field\_name\_cond := COALESCE(

TRIM(

LOWER(

replace

(

(json\_parameters -> 'field\_name\_cond') ::text,

'"',

''

)

)

),

''

);

if(

value\_country = ''

or value\_country = 'all'

) then slqConditionCountry := '';

else slqConditionCountry := ' lower(country) =''' || value\_country || '''';

end if;

if (value\_table\_name <> '') then

/\*\*lookup with table name defined by user\*\*/

if (strpos(value\_field\_name, '\_multi') > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_data\_multi(value\_table\_name, value\_single, value\_country);

elsif (strpos(value\_field\_name, 'month') > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_all\_month\_standard(value\_single, value\_country, value\_table\_name);

elsif (strpos(value\_field\_name, 'gender') > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_gender\_standard(value\_table\_name, value\_single, value\_country);

elsif (

value\_single = ''

or value\_single = 'all'

) then sqlLK := 'SELECT array\_to\_json(ARRAY(SELECT (' || value\_field\_name || ') FROM ' || value\_table\_name || '';

if(slqConditionCountry <> '') then sqlLK := sqlLK || ' WHERE ' || slqConditionCountry;

end if;

else sqlLK := 'SELECT array\_to\_json(ARRAY(SELECT (' || value\_field\_name || ') FROM ' || value\_table\_name || ' WHERE ' || value\_field\_name || ' like ''' || value\_single || '%''';

if(slqConditionCountry <> '') then sqlLK := sqlLK || ' AND ' || slqConditionCountry;

end if;

end if;

if(sqlLK <> '') then sqlLK := sqlLK || ' ORDER BY id LIMIT 1000)) as single\_words';

end if;

--RAISE INFO 'sql: %', sqlLK;

if(sqlLK <> '') then

EXECUTE

(sqlLK) INTO result\_single;

end if;

if(rcResult is not null) then FETCH rcResult INTO v\_f,

v\_s;

WHILE

FOUND

LOOP

result\_single := v\_f;

if(v\_s is not null) then result\_multi := v\_s;

--RAISE INFO 'v\_s: %', v\_s;

end if;

FETCH rcResult INTO v\_f,

v\_s;

END

LOOP

;

CLOSE rcResult;

end if;

RETURN

QUERY

SELECT

result\_single,

result\_multi;

else if((strpos(value\_field\_name, 'surname')) > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_surname\_json\_giangtm(value\_single, value\_multi);

elsif((strpos(value\_field\_name, 'given')) > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_given\_only\_json\_giangtm(value\_single, value\_multi);

elsif (strpos(value\_field\_name, 'gender') > 0) then OPEN rcResult FOR

select

\*

from

sp\_lookup\_gender\_standard(value\_table\_name, value\_single, value\_country);

elsif(value\_field\_name = 'symbol-group') then if (

value\_single = ''

or value\_single = 'all'

) then sqlLK := 'SELECT json\_agg(symbol\_group order by id) FROM tbl\_symbol\_group';

else sqlLK := 'SELECT json\_agg(symbol\_group ORDER BY id) FROM tbl\_symbol\_group WHERE symbol\_original = ' || quote\_literal(trim(lower(value\_single)));

end if;

--RAISE INFO 'sql: %', sqlLK;

EXECUTE

(sqlLK) INTO result\_single;

end if;

if(rcResult is not null) then FETCH rcResult INTO v\_f,

v\_s;

WHILE

FOUND

LOOP

result\_single := v\_f;

if(v\_s is not null) then result\_multi := v\_s;

--RAISE INFO 'v\_s: %', v\_s;

end if;

FETCH rcResult INTO v\_f,

v\_s;

END

LOOP

;

CLOSE rcResult;

end if;

RETURN

QUERY

SELECT

result\_single,

result\_multi;

end if;

END;

$function $