Artículo
Dani Fibla · 21 dic, 2020 Lectura de 4 min

Cómo producir JSON desde SQL

Recientemente publiqué varias actualizaciones sobre nuestras funciones JSON y estoy encantado de que muchos hayáis hecho comentarios. Hoy, me gustaría centrarme en otra faceta: Producir JSON con una consulta SQL.

SQL es un concepto muy importante para recuperar registros de tu modelo relacional. Imagina que quieres consultar datos y mostrarlos como una sencilla estructura JSON en un servicio REST. Generalmente, debes consultar tus datos, después iterar sobre el conjunto de resultados y, finalmente, construir una estructura JSON para cada registro. Debes escribir un código personalizado.

Hemos añadido funciones SQL estándar para que sea más fácil producir JSON directamente de una consulta SQL sin escribir código: JSONOBJECT y JSONARRAY . Estas dos funciones son nuevas en Caché 2016.2.

Considera la siguiente tabla Baking.Pastry para los ejemplos:

Fila	Tipo	Descripción
1	Pasta choux	Una pasta li crema
2	Pasta hojaldre	Muchas cap se infle cuar
3	Pasta filo	Varias capa envuelve al

JSONOBJECT

JSONOBJECT es una función que toma varios pares clave-valor y produce un objeto JSON.

```
SELECT JSON_OBJECT('pastry_type' : Type, 'pastry_description' : Description) AS pastr
yJSON FROM Baking.Pastry

Row    pastryJSON

----   ---------------

1      {"pastry_type" : "Choux Pastry", "pastry_description" : "A light pastry that i
s often filled with cream"}

2      {"pastry_type" : "Puff Pastry", "pastry_description" : "Many layers that cause
 the pastry to expand or puff when baked"}

3      {"pastry_type" : "Filo Pastry", "pastry_description" : " Multiple layers of a
paper-thin dough wrapped around a filling"}
```

En este ejemplo, la función JSONOBJECT produce un objeto JSON para cada registro. Cada objeto contiene dos propiedades pastrytype y pastrydescription , ya que proporcionamos dos argumentos a la función. Cada argumento consta de dos partes, delimitadas por dos puntos:

1. El nombre de la clave que debe introducirse en el objeto
2. El valor asociado con esa clave

Este ejemplo asigna claves estáticas porque solo proporcioné una cadena literal, por ejemplo: 'pastrytype' . Para el valor, hago referencia a una columna, por ejemplo: Type y cualquiera que sea el contenido de esa columna, se establecerá como el valor de la clave asociada. Este es un caso de uso común para construir un objeto JSON, pero pasando a una columna la clave, también se pueden crear claves dinámicamente.

JSONARRAY

JSONARRAY funciona de forma similar. Construye una matriz JSON y cada argumento que se transfiere recibirá el valor correspondiente de la matriz.

```
SELECT JSON_ARRAY(Type, Description) AS pastryJSON FROM Baking.Pastry


Row  pastryJSON

---- ---------------

1    ["Choux Pastry" , "A light pastry that is often filled with cream"]

2    ["Puff Pastry" , "Many layers that cause the pastry to expand or puff when baked
"]

3    ["Filo Pastry" , "Multiple layers of a paper-
thin dough wrapped around a filling"]
```

JSONARRAY es una función bastante sencilla. En este ejemplo se crea una matriz, que guarda dos elementos en cada fila. El primer elemento se completa con el valor de la columna Tipo, mientras que el segundo elemento se completa con el valor de la columna Descripción.

Escenarios avanzados

Tal vez tengas la obligación de crear una estructura JSON más compleja. Un argumento de valor puede ser llamar a una función anidada JSONARRAY o JSONOBJECT , que te permitirá construir estructuras JSON anidadas. Vamos a tomar el primer ejemplo y envolver el objeto JSON en una estructura de encabezado:

```
SELECT JSON_OBJECT('food_type' : 'pastry', 'details' : JSON_OBJECT('type' : Type, 'de
scription' : Description)) AS pastryJSON FROM Baking.Pastry


Row  pastryJSON

---- ---------------

1    {"food_type" : "pastry", "details" : {"type" : "Choux Pastry", "description" : "
A light pastry that is often filled with cream"}}

2    {"food_type" : "pastry", "details" : {"type" : "Puff Pastry", "description" : "
Many layers that cause the pastry to expand or puff when baked"}}

3    {"food_type" : "pastry", "details" : {"type" : "Filo Pastry", "description" : "
Multiple layers of a paper-thin dough wrapped around a filling"}}
```

Hay más funciones JSON en SQL que planeamos implementar en versiones posteriores, pero estas dos son un buen comienzo. El caso de uso principal es construir elementos JSON sencillos, a partir de tus datos relacionales,

sin necesidad de escribir código. Esto te permitirá publicar JSON desde un sistema, incluso si no puedes cambiar el backend.

Para crear estructuras más complejas, es más eficiente elaborarlas con la nueva interfaz de composición, que permite transformar un objeto persistente/registrado en un objeto/matriz dinámica. Después, se puede modificar fácilmente la estructura como se crea más conveniente, y finalmente se puede exportar una cadena JSON mediante una llamada $toJSON(). Trataré este tema con mayor profundidad en una publicación futura.

#JSON #SQL #Caché