JSON数据计算与入库

目 录

CONTENTS

01

导入与解析

1. 单层的JSON数据文件

我们先从一个简单的例子入手，看看普通键值映射的JSON文件如何读取。
下面是产品信息的JSON数据：

[{"PRODUCT_ID":1,"PRODUCT_NAME":"Apple Juice",
"SUPPLIER_ID":2,"CATEGORY_ID":1, ...},
{"PRODUCT_ID":2,"PRODUCT_NAME":"Milk",
"SUPPLIER_ID":1,"CATEGORY_ID":1, ...},
{"PRODUCT_ID":3,"PRODUCT_NAME":"Tomato sauce",
"SUPPLIER_ID":1,"CATEGORY_ID":2, ...},
{"PRODUCT_ID":4,"PRODUCT_NAME":"Salt",
"SUPPLIER_ID":2,"CATEGORY_ID":2, ...},
...]

1. 单层的JSON数据文件

SPL导入JSON数据文件只需要简单的一句脚本：

```
=json(file("product.json").read())
```

执行结果如下：

PRODUCT_ID	PRODUCT_NAME	SUPPLIER_ID	CATEGORY_ID	…
1	Apple Juice	2	1	…
2	milk	1	1	…
3	Tomato sauce	1	2	…
4	salt	2	3	…
…	…	…	…	…

2. 明细数据相同结构的多层JSON数据文件

下面是订单信息的JSON数据。分为两层: 第一层是国家和地区，第二层是明细数据。现在我们想要导入中国华北和华南地区2013年的订单。


```
[{"COUNTRY":"China","AREA":"Northeast China","ORDERS":[
{"ORDER_ID":10252,"CUSTOMER_ID":"SUPRD","EMPLOYEE_ID":4, ...},
{"ORDER_ID":10318,"CUSTOMER_ID":"ISLAT","EMPLOYEE_ID":8, ...},
...]},
{"COUNTRY":"China","AREA":"East China","ORDERS":[
{"ORDER_ID":10249,"CUSTOMER_ID":"TOMSP","EMPLOYEE_ID":6, ...},
{"ORDER_ID":10251,"CUSTOMER_ID":"VICTE","EMPLOYEE_ID":3, ...},
...]},
...]
```

2. 明细数据相同结构的多层JSON数据文件

定义参数：Country、Area和Year。以后导入不同国家、地区和年份的时候，就不再需要修改SPL，只需要修改相应的参数值就行了。这里需要注意的是，Area的值是序列，这样就可以同时读取多个地区的数据。如下图：

Name	Value
Country	China
Area	[North China, South China]
Year	2013

2. 明细数据相同结构的多层JSON数据文件

我们先看一下SPL脚本：

	A	B
1	=json(file("orders.json").read())	=A1.select(COUNTRY==Country && Area.contain(AREA))
2	=B1.news(ORDERS;COUNTRY, AREA,${B1.ORDERS.fname().concat@c()})	=A2.select(year(ORDER_DATE)==Year)

2. 明细数据相同结构的多层JSON数据文件

```
=json(file("orders.json").read())
```

首先导入JSON文件，数据是多层的：

COUNTRY	AREA	ORDERS
China	Northeast China	[[10252,SUPRD,4,...],[10315,ISLAT,4,...],...]
China	East China	[[10249,TOMSP,6,...],[10251,VICTE,3,...],...]
China	Central China	[[10254,CHOPS,5,...],[10265,BLONP,2,...],...]
China	North China	[[10248,VINET,5,...],[10250,HANAR,4,...],...]
China	South China	[[10287,RICAR,8,...],[10296,LILAS,6,...],...]

双击可以查看明细数据

ORDER_ID	CUSTOMER_ID	EMPLYEE_ID	…
10287	RICAR	8	…
10296	LILAS	6	…
…	…	…	…

2. 明细数据相同结构的多层JSON数据文件

年份和地区字段就在第一层，可以直接筛选出中国华北和华南的数据。

```
=A1.select(COUNTRY==Country && Area.contain(AREA))
```

结果如下：

COUNTRY	AREA	ORDERS
China	North China	[[10248,VINET,5,...],[10250,HANAR,4,...],...]
China	South China	[[10287,RICAR,8,...],[10296,LILAS,6,...],...]

2. 明细数据相同结构的多层JSON数据文件

```
=B1.news(ORDERS;COUNTRY, AREA,${B1.ORDERS.fname().concat@c()})
```

用过滤后的结果生成序表，由国家、地区和订单明细的字段组成。结果如下：

COUNTRY	AREA	ORDER_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_DATE
China	North China	10248	VINET	5	2012-07-04
China	North China	10250	HANAR	4	2012-07-08
China	North China	10253	HANAR	3	2012-07-10
China	North China	10255	RICSU	9	2012-07-12

在这里，news函数的参数使用了宏。宏用${}把表达式括起来，SPL在计算时会先计算宏表达式，再将计算结果作为字符串值替换${}。A2实际执行的是：=B1.news(ORDERS;COUNTRY, AREA, ORDER_ID, CUSTOMER_ID, EMPLOYEE_ID, ORDER_DATE, …)

2. 明细数据相同结构的多层JSON数据文件

最后，从序表中选出订单日期的年份是2013的记录。

```
=A2.select(year(ORDER_DATE)==Year)
```

最终结果如下：

COUNTRY	AREA	ORDER_ID	CUSTOMER_ID	EMPLOYEE_ID	ORDER_DATE
China	North China	10402	ERNSH	8	2013-01-02
China	North China	10403	ERNSH	4	2013-01-03
China	North China	10404	MAGAA	2	2013-01-03
China	North China	10407	OTTIK	2	2013-01-07

3. 明细数据不同结构的多层JSON数据文件

因为数据来源的复杂性，JSON数据文件的明细数据有可能是不同结构的。下面的销售数据中：第一层以年和月为维度，第二层以国家为维度，第三层是明细数据。但是明细数据中，由于销售渠道不同，数据结构是不完全一致的。现在我们要读取2017和2018年美国和加拿大的销售数据。


```
[{"YEAR":2016,"MONTH":1,"SALES":
[{"COUNTRY":"Germany","SALES":
[{"ORDERNUMBER":10101,"QUANTITYORDERED":25,"PRICEEACH":100,"ORDE
RLINENUMBER":4,"SALES":3782,"ORDERDATE":"1/9/2016 0:00", ...}, ...], ...],
{"YEAR":2016,"MONTH":2,"SALES":
[{"COUNTRY":"Denmark","SALES":
[{"ORDERNUMBER":10105,"QUANTITYORDERED":50,"PRICEEACH":100,"ORDE
RLINENUMBER":2,"SALES":7208,"ORDERDATE":"2/11/2016 0:00", ...}, ...], ...],
...]
```

3. 明细数据不同结构的多层JSON数据文件

我们首先要确定明细数据的结构。本例中我们想要列出全部的字段，明细数据不包含该字段时设置为空。例如下面数据中缺少ADDRESSLINE2字段：

YEAR	COUNTRY	ORDERNUMBER	ADDRESSLINE1	ADDRESSLINE2
2017	USA	10353	2440 Pompton St.	
2017	USA	10352	16780 Pompton St.	

为了使用方便，我们还是先定义两个参数：Year和Country：

Name	Value
Year	[2017, 2018]
Country	[USA, Canada]

3. 明细数据不同结构的多层JSON数据文件

接下来先看一下SPL：

	A	B
1	=json(file("sales.json").read())	=A1.select(Year.contain(YEAR))
2	=B1.news(SALES;YEAR,MONTH,COUNTRY,SALES)	=A2.select(Country.contain(COUNTRY))
3	for(B2)	=A3.SALES.fname()&B3
4	=B2.news(SALES; YEAR, COUNTRY,${B3.concat@c()})	

3. 明细数据不同结构的多层JSON数据文件

=json(file("sales.json").read())	=A1.select(Year.contain(YEAR))

首先还是导入多层的JSON文件。由于年份字段就在第一层，可以直接筛选出2017和2018年份的数据：

YEAR	MONTH	SALES
2017	1	[[France,[10211,41,100, ...],[10211,41,100, ...], ...], ...]
2017	2	[[Australia,[10223,37,100, ...],[10223,47,100, ...], ...], ...]
2017	3	[[Australia,[10227,25,100, ...],[10227,31,48.52, ...], ...], ...]
2017	4	[[Canada,[10235,24,76.03, ...],[10235,23,96.29, ...], ...], ...]
2017	5	[[Finland,[10247,44,100, ...],[10247,25,100, ...], ...], ...]

3. 明细数据不同结构的多层JSON数据文件

```
=B1.news(SALES;YEAR,MONTH,COUNTRY,SALES)
```

使用news函数，用来把年月字段和下一层的国家和月销售明细拼在一起：

YEAR	MONTH	COUNTRY	SALES
2017	1	France	[[10211,41,100, ...],[10211,41,100, ...], ...]
2017	1	Japan	[[10210,23,100, ...],[10210,34,100, ...], ...]
2017	1	Spain	[[10212,39,100, ...],[10212,33,100, ...], ...]
2017	1	UK	[[10213,38,94.79, ...],[10213,25,83.39, ...], ...]
2017	1	USA	[[10215,35,100, ...],[10209,39,100, ...], ...]

3. 明细数据不同结构的多层JSON数据文件

B2：通过A2.select(Country.contain(COUNTRY))从中筛选出来美国和加拿大的数据。
A3~A4：由于明细数据可能结构有所不同，我们使用全量的字段名作为参数来创建序表。字段的值会根据名称设置，无此字段的会缺省为空值（例如下图"ADDRESSLINE1"和" ADDRESSLINE2"字段）：

YEAR	COUNTRY	ORDERNUMBER	ADDRESSLINE1	ADDRESSLINE2
2017	USA	10353	2440 Pompton St.	
2017	USA	10352	16780 Pompton St.	
2017	USA	10352	16780 Pompton St.	
2018	USA	10369		
2018	USA	10362		
2018	USA	10371		

3. 明细数据不同结构的多层JSON数据文件

执行后可以看到最终结果：

YEAR	COUNTRY	ORDERNUMBER	QUANTITYORDERED	PRICEEACH	ORDERLINENUMBER
2017	USA	10215	35	100	3
2017	USA	10209	39	100	8
2017	USA	10215	46	100	2
2017	USA	10215	27	89.38	10
2017	USA	10215	33	43.13	9
2017	USA	10215	49	100	4

至此，一个多层结构的明细数据结构不完全一致的JSON文件，就展开成一个二维表了。

02

序表入库

1. 单表入库

以1.1中产品订单信息的JSON文件为例，要把解析后的序表更新到数据库的Product表中。

JSON文件：

[{"PRODUCT_ID":1,"PRODUCT_NAME":"Apple Juice", "SUPPLIER_ID":2,"CATEGORY_ID":1, ...},
{"PRODUCT_ID":2,"PRODUCT_NAME":"Milk", "SUPPLIER_ID":1,"CATEGORY_ID":1, ...},
{"PRODUCT_ID":3,"PRODUCT_NAME":"Tomato sauce", "SUPPLIER_ID":1,"CATEGORY_ID":2, ...},
...]

数据库表：

Product
PRODUCT_ID
PRODUCT_NAME
SUPPLIER_ID
CATEGORY_ID
...

1. 单表入库

SPL处理序表入库非常简单，使用db.update()函数即可。SPL脚本如下：

	A
1	=json(file("product.json").read())
2	=connect("db")
3	=A2.update(A1, Product)

A1：导入JSON文件为序表。A2：连接数据源

A3：使用db.update()函数，把A1导入的序表，更新到数据库的Product表中。需要注意的是，这里省略了update函数的主键参数。此时会按照数据库表Product的主键更新；如果Product表没有主键，则会按照A1的主键更新；都没有主键时会按照第一个字段更新。

2. 多表入库

下面以订单信息的JSON文件为例。JSON数据分为两层: 第一层是订单，第二层是订单明细。要把2018年及以后的订单和订单明细数据，分别更新到数据库的订单表和订单明细表中。

[{"ORDER_ID":10248,"ORDER_DATE":"2012-07-04",...,"ORDER_DETAILS":[
{"PRODUCT_ID":17,"PRICE":14,"AMOUNT":12, ...},
{"PRODUCT_ID":42,"PRICE":9,"AMOUNT":9, ...},
...]}, ...]

Order
ORDER_ID
ORDER_DATE
CUSTOMER_ID
EMPLOYEE_ID
...

1 : N

OrderDetail
ORDER_ID
PRODUCT_ID
PRICE
AMOUNT
...

2. 多表入库

先来看一下SPL，如下：

	A	B
1	=json(file("orders.json").read())	=A1.select(year(ORDER_DATE)>=2018)
2	=connect("demo")	
3	=B1.fname().delete(B1.fname().len())	=A2.update(B1,Order,${A3.concat@c()})
4	=B1.conj(ORDER_DETAILS.derive(B1.ORDER_ID:ORDER_ID))	=A2.update(A4,OrderDetail)

2. 多表入库

```
=json(file("orders.json").read())
```

首先导入JSON文件，数据是两层的：

ORDER_ID	ORDER_DATE	CUMSTOMER_ID	ORDER_DETAILS
10248	2012-07-04	VINET	[[17,14,12,...],[42,9,10,...],...]
10249	2012-07-05	TOMSP	[[14,18,9,...],[51,42,4,...],...]
10250	2012-07-08	HANAR	[[41,7,10,...],[51,42,3,...],...]
10251	2012-07-08	VICTE	[[22,16,6,...],[57,15,15,...],...]
10252	2012-07-09	SUPRD	[[20,64,40,...],[33,2,25,...],...]

双击可以查看明细数据

PRODUCT_ID	PRICE	AMOUNT	…
20	64	40	…
33	2	25	…
…	…	…	…

2. 多表入库

订购日期字段就在第一层，可以直接筛选出2018年及以后的数据。

```
=A1.select(year(ORDER_DATE)>=2018)
```

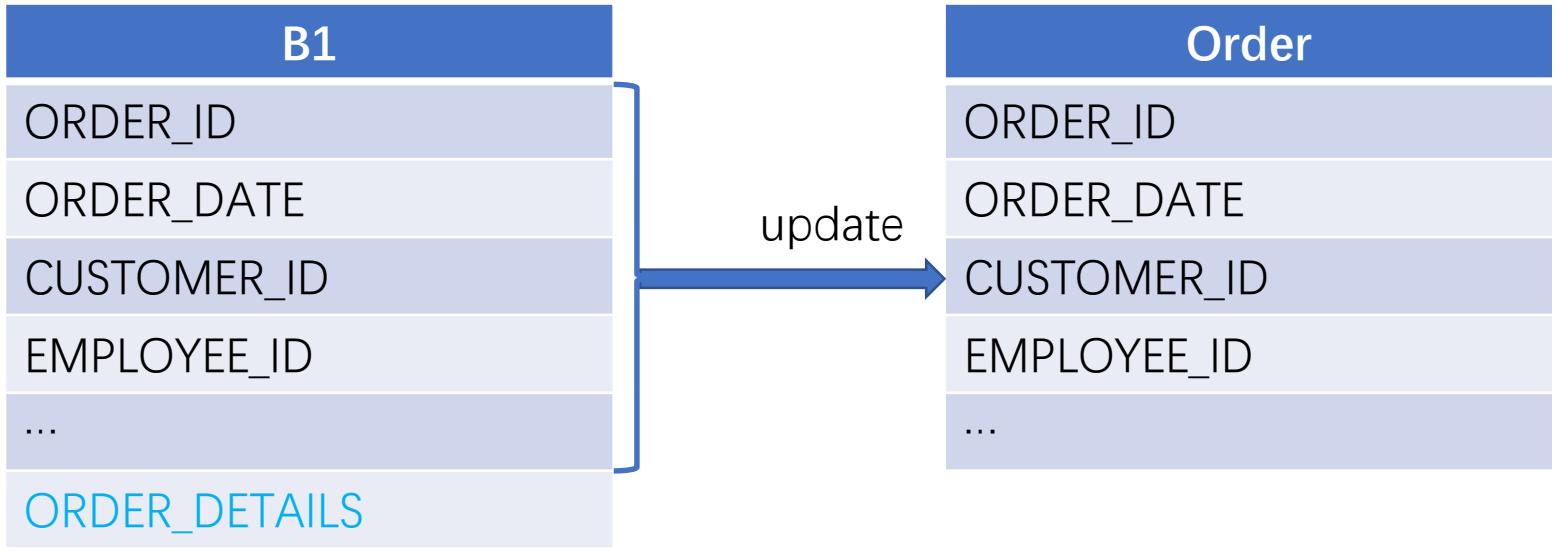
结果如下：

ORDER_ID	ORDER_DATE	CUMSTOMER_ID	ORDER_DETAILS
10808	2018-01-01	OLDWO	[[56,38,20,...],[76,18,50,...]]
10809	2018-01-01	WELLI	[[52,7,20,...]]
10810	2018-01-01	LAUGB	[[13,6,7,...],[25,14,5,...],...]

2. 多表入库

=B1.fname().delete(B1.fname().len())	=A2.update(B1,Order,${A3.concat@c()})

连接数据库以后，首先更新主表订单表。但是B1比数据库表多了一个ORDER_DETAILS字段，使用update函数更新时需要指定更新字段。字段参数使用了宏，通过B1.fname()来获取所有字段名，再删除最后一个成员即可。



2. 多表入库

```
=B1.conj(ORDER_DETAILS.derive(B1.ORDER_ID:ORDER_ID))
```

首先用derive()函数，将订单ID字段添加到ORDER_DETAILS中。再使用conj()函数，将每个订单的ORDER_DETAILS展开拼在一起，就与数据库中的订单明细表的数据结构一致了。结果如下：

ORDER_ID	ORDER_DETAILS
10808	[[56,38,20,...],[76,18,50,...]]
10809	[[52,7,20,...]]

derive

ORDER_ID	ORDER_DETAILS				
10808	ORDER_ID	PRICE	AMOUNT	DISCOUNT	ORDER_ID
	56	38	20	0.15	10808
	76	18	50	0.15	10808
10809	ORDER_ID	PRICE	AMOUNT	DISCOUNT	ORDER_ID
	52	7	20	0	10809

B1.conj(ORDER_DETAILS)

ORDER_ID	PRICE	AMOUNT	DISCOUNT	ORDER_ID
56	38	20	0.15	10808
76	18	50	0.15	10808
52	7	20	0	10809

2. 多表入库

```
=A2.update(A4,OrderDetail)
```

最后，更新子表订单明细表。由于A4的数据结构与数据库表一致，不再需要指定更新字段。

A4
ORDER_ID
PRODUCT_ID
PRICE
AMOUNT
DISCOUNT

update →

OrderDetail
ORDER_ID
PRODUCT_ID
PRICE
AMOUNT
DISCOUNT

03

总结

总结

使用JSON数据的流程

首先读入JSON文件，再利用json()函数解析为序表。

根据实际需求，对数据进行解析与计算。SPL的序表提供了丰富的函数，可以胜任各种运算。

序表入库只要使用db.update()函数即可。JSON文件对应多个表时，要注意更新的顺序，从主表到子表。

从前面章节我们可以看到，JSON数据使用的重点，在于解析与计算部分。SPL在应对多层、不同构的复杂数据时，可以简单的用“表.字段”来引用成员，还有丰富的函数支持计算。

THANKS

感谢聆听 批评指导