转置

目 录

CONTENTS

01

行转列

数据库的pivot

下面是学生成绩表。统计每个班的各科最高分，按列显示。

Class	StudentID	Subject	Score
Class one	1	Math	89
Class one	1	Chinese	93
Class two	2	Math	92
Class two	2	Chinese	97

数据库的pivot函数支持行转列

Class	MathMax	ChineseMax
Class one	89	93
Class two	92	97

数据库的pivot

SQL语句如下：

```
select * from ( select Class, Subject, Score from StudentScore )
pivot (
    max(Score) for Subject in (
        'Math' as MathMax, 'Chinese' as ChineseMax
    )
)
```

这里是以oracle为例。并不是所有数据库都有pivot函数，只有主流数据库的较新版本中，才开始支持pivot。

SPL的pivot

首先从数据库中取出各科的最高分。

Class	Subject	MaxScore
Class one	Math	89
Class one	Chinese	93
Class two	Math	92
Class two	Chinese	97

再使用SPL的pivot函数行转列。

Class	MathMax	ChineseMax
Class one	89	93
Class two	92	97

SPL的pivot

SPL脚本如下：

	A
1	=connect("oracle")
2	=A1.query("select Class, Subject, max(Score) MaxScore from StudentScore group by Class, Subject")
3	=A2.pivot(Class; Subject, MaxScore; "Math":"MathMax", "Chinese":"ChineseMax")

A1：连接数据库

A2：从数据库取数，这里直接聚合计算了各科最高分。

A3：使用pivot函数实现行转列。

行转列小结

不是所有数据库都支持pivot。

例如MySQL就没有pivot的函数，需要使用子查询分组汇总，再left join等方式实现行转列。

SPL的pivot适应性更广

行转列后还需要列间计算的情况，SQL语句很难处理。多数据来源的情况SQL无法解决。

02

列转行

SQL的unpivot

下面是学生成绩表。要查询学生的最高分。

StudentID	Math	Chinese
1	89	93
2	92	97
3	91	88

数据库的unpivot函数支持列转行

StudentID	BestScore
1	93
2	97
3	91

SQL的unpivot

还是以oracle为例，SQL语句如下：

```
With
T1 as (select * from StudentScore unpivot (Score for Subject in
(Math, Chinese)))
select StudentID, max(T1.Score) BestScore from T1 group by
StudentID
```

SPL的列转行

SPL的列转行看起来更加清晰。第一步列转行，第二步分组汇总取每个学生的最高分。

StudentID	Math	Chinese
1	89	93
2	92	97
3	91	88

StudentID	Subject	Score
1	Math	89
1	Chinese	93
2	Math	92
2	Chinese	97
3	Math	91
3	Chinese	88

StudentID	BestScore
1	93
2	97
3	91

SPL的列转行

SPL脚本如下：

	A
1	=connect("oracle")
2	=A1.query("select * from StudentScore")
3	=A2.pivot@r(StudentID; Subject, Score; Math:"Math", Chinese:"Chinese")
4	=A3.groups(StudentID; max(Score):BestScore)

A1：连接数据库。A2：读取学生成绩表。

A3：使用pivot函数的@r选项实现列转行。

A4：对转置后的数据，按学号进行分组后取最大值。

03

双向转置

双向转置

按渠道分类的销售表，按日期记录，如下图：

Day	Online	Store
20190101	2400	1863
20190102	1814	670
20190103	3730	1444

想要转换成如下结果：

Category	20190101	20190102	20190103
Online	2400	1814	3730
Store	1863	670	1444

双向转置

首先列转行，将Online和Store转换为类别列的字段值。

Day	Online	Store
20190101	2400	1863
20190102	1814	670
20190103	3730	1444

Day	Category	Sales
20190101	Online	2400
20190101	Store	1863
20190102	Online	1814
20190102	Store	670
20190103	Online	3730
20190103	Store	1444

然后行转列，将Day字段值的唯一值转换为列名。

Category	20190101	20190102	20190103
Online	2400	1814	3730
Store	1863	670	1444

双向转置

SPL脚本如下：

	A
1	=connect("db")
2	=A1.query("select * from Sales")
3	=A2.pivot@r(Day; Category, Sales)
4	=A3.pivot(Category; Day, Sales)

A3：使用pivot@r列转行，将Online，Store等渠道类型转换为Category的字段值。

A4：使用pivot行转列，将日期字段的值转换为列。

pivot函数小结

pivot函数适合静态转置

当行列数不发生变化时，直接使用pivot函数可以很方便的实现转置。当需要双向转置时，SPL的pivot和pivot@r函数可以解决。

面对转置后表结构无法确定的动态转置，pivot函数就无法胜任了。我们在后面章节将会给出解决方案。

04

动态行列转置

1.pivot函数自动生成列

有一个员工表，记录了员工的部门、所在地和收入等信息，如下图：

Name	Dept	Area	Salary
David	Sales	Beijing	8000
Daniel	R&D	Beijing	15000
Andrew	Sales	Shanghai	9000
Robert	Sales	Beijing	26000
Rudy	R&D	Shanghai	23000

统计各部门在不同地区的平均工资，现在不知道有哪些地区，想要转换成如下结果：

Dept	Beijing	Shanghai	…
Sales	13000	11000	…
R&D	15000	14000	…

1.pivot函数自动生成列

这个例子是行转列，目标列需要从数据中提取。pivot函数支持半动态转置，不指明目标列（源列）时，它会自动找所有没放在分组中的列。

SPL脚本如下：

	A
1	=connect("db")
2	=A1.query("select Dept,Area,avg(Salary) as AvgSalary from Employee group by Dept,Area")
3	=A2.pivot(Dept; Area, AvgSalary)

A1：连接数据源。

A2：从员工表中取出按部门、地区分组的平均工资。

A3：使用pivot函数行转列，这里省略了目标列。

2.动态行列转置

有一个记录收入情况的个人收入表，如下图：

Name	Source	Income
David	Salary	8000
David	Bonus	15000
Daniel	Salary	9000
Andrew	Shares	26000
Andrew	Sales	23000
Robert	Bonus	13000

每个人的收入来源都可能不相同，想要转换成如下结果：

Category	Source1	Income1	Source2	Income2
David	Salary	8000	Bonus	15000
Daniel	Salary	9000		
Andrew	Shares	26000	Sales	23000
Robert	Bonus	13000		

2.动态行列转置

我们不确定行转列后，列的数量，甚至连列名也不能完全确定。这时就不能使用pivot 函数了，而需要使用动态转置的方法。

SPL脚本如下：

	A	B
1	=connect("db")	=A1.query("select * from Income")
2	=B1.group(Name)	=A2.max(~.len())
3	=create(Name, ${B2.("Source"+string(~)+", Income"+string(~)).concat@c()})	
4	for A2	=A4. Name | A4.conj([Source, Income])
5		>A3.record(B4)

A3：根据分组后，数量最多的组成员确定列数，并动态产生列名，创建序表。

A4~B5：循环分组后的收入表成员，将每组的姓名、收入来源和收入金额拼在一起，添加到A3创建的序表中。

动态行列转置小结

动态转置先计算目标数据结构

动态转置的关键是，先计算出目标数据结构。确定了表结构以后，再把数据按照数据结构拼成一条一条的记录，插入到目标表中。这种思路对于有些静态转置也同样适用。

3. 复杂静态行列转置

我们再来看一个例子，有一个记录日常考勤信息的表，如下图：

Per_Code	in_out	Date	Time	Type
1110263	1	2013-10-11	09:17:14.0000000	In
1110263	6	2013-10-11	11:37:00.0000000	Break
1110263	5	2013-10-11	11:38:21.0000000	Return
1110263	0	2013-10-11	11:43:21.0000000	NULL
1110263	6	2013-10-11	13:21:30.0000000	Break
1110263	5	2013-10-11	14:25:58.0000000	Return
1110263	2	2013-10-11	18:28:55.0000000	Out

每七条数据为一组，想要转换成如下结果：

Per_Code	Date	In	Out	Break	Return
1110263	2013-10-11	9:17:14	18:28:55	11:37:00	11:38:21
1110263	2013-10-11	9:17:14	18:28:55	13:21:30	14:25:58

3. 复杂静态行列转置

虽然转置后的表结构是可以确定的，但是用pivot实现起来会很复杂。这时可以先创建目标数据结构，再往里面填数据。

SPL脚本如下：

	A	B
1	=connect("db").query("select * from DailyTime order by Per_Code,Date,Time")	=A1.group(Per_Code,Date)
2	=create(Per_Code,Date,In,Out,Break,Return)	=B1.([1,7,2,3,1,7,5,6].(B1.~(~)))
3	=B2.conj([~.Per_Code,~.Date]|~.(Time).m([1,2,3,4])|[~.Per_Code,~.Date]|~.(Time).m([5,6,7,8]))	>A2.record(A3)

A1：查询数据并按人员编号、日期和时间分组。B1：按人员编号和日期分组。

A2：创建一个存放最后结果的空序表。

B2：对每个组，依次取出第 1,7,2,3,1,7,5,6 条记录，这就是有序的全天记录。

A3：将每条记录的数据全部整理到一个序列中。B2：将记录添加到A2创建的序表中。

4. 复杂动态行列转置

有两个表，分别是用户表和记录表，表示用户在某日存在一条某活动的记录。两表通过用户ID关联。如下图：

User
ID
Name

Record
ID
Date

想要显示2018年的每周，用户是否有活动的记录，如下表：

Week	User1	User2	User3
1	Yes	No	Yes
2	Yes	Yes	No
3	Yes	No	Yes
4	No	Yes	Yes

4. 复杂动态行列转置

这个例子看起来复杂一些，但是思路不变。还是先创建目标数据结构，再往里面填数据。
SPL脚本如下：

	A	B
1	=connect("db").query("select t1.ID as ID, t1.Name as Name, t2.Date as Date from User t1, Record t2 where t1.ID=t2.ID")	
2	=A1.derive(interval@w("2018-01-01",Date)+1:Week)	=A2.max(Week)
3	=A2.group(ID)	=B2.new(~:Week,${A3.("\"No\":"+Name).concat@c()})
4	=A3.run(~.run(B3(Week).field(A3.#+1,"Yes")))	

A1：查询用户表和记录表，按用户ID连接。A2：根据日期计算出周序号。
B2：找到最大的周序号。A3：按照用户ID进行分组。
B3：根据最大的周序号创建一个空序表，赋予缺省值"No"。
A4：每组每条数据，通过周序号在目标表中定位到对应记录，替换该用户值为"Yes"。

05

转置同时列间计算

转置同时列间计算

行列转换的同时往往伴随列间计算，例如下面数据：

ID	customID	name	amount_payable	due_date	amount_paid	pay_date
112101	C013	CA	12800	2014-02-21	12800	2014-12-19
112102	C013	CA	3500	2014-06-15	3500	2014-12-15
112103	C013	CA	2600	2015-03-21		

要求根据指定年份（如2014），输出每月应付金额，若无当月数据，则当月应付金额为上月该值。

name	1	2	3	4	5	6	7	8	9	10	11	12
CA		12800	12800	12800	12800	3500	3500	3500	3500	3500	3500	3500
…												

转置同时列间计算

运算过程仍然是先产生空结果集后追加数据，不同的是，这里要追加的数据需要经常一系列计算才能得到。SPL脚本如下：

	A	B
1	=connect("db").query("select * from Payment.txt where year(due_date)=2014")	
2	=create(name,${12.().string@d()})	=A1.group(customID)
3	for B2	=12.(null)
4		>A3.run(B3(month(due_date))= amount_payable)
5		>B3.run(~=ifn(~,~[-1]))
6		=A2.record(B2.name|B3)

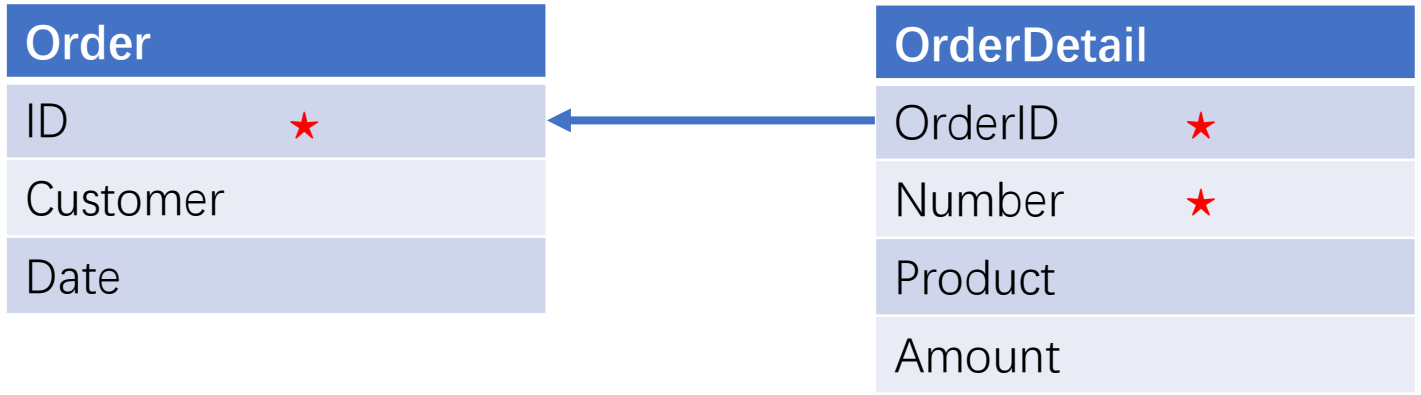
A1：查询2014年数据。A2：生成带有12个月的结果空序表。A3：按customID分组。A3~B6：循环分组，B4设置相应月份的应付金额，B5将空值置为前一个月的数值，B6将记录插入结果序表中。

06

表间关联列转行

1.子表动态插入主表

订单表和订单明细表是主子表关系，每个订单有多条明细数据。如下图：



订单明细表中每个订单的明细数据是不定长的。想要查询出如下表格：

ID	Customer	Date	Product1	Amount1	Product2	Amount2	Product3	Amount3
1	3	20190101	Apple	5	Milk	3	Salt	1
2	5	20190102	Beef	2	Pork	4		
3	2	20190102	Pizza	3				

1.子表动态插入主表

SPL脚本如下：

	A	B
1	=connect("db") .query("select * from OrderDetail left join Order on Order.ID=OrderDetail.OrderID")	
2	=A1.group(ID)	=A2.max(~.count()).("Product"+string(~)+","+"Amount"+string(~)).concat@c()
3	=create(ID,Customer,Date,${B2})	>A2.run(A3.record([ID,Customer,Date]|~.([Product,Amount]).conj()))

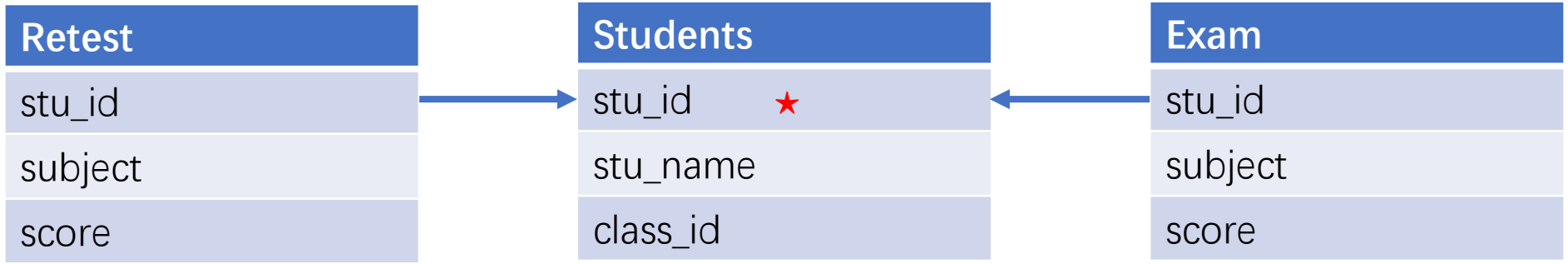
A1：将订单表和订单明细表连接取数。A2：按订单ID分组。

B2~A3：根据分组内成员数量最多的组，动态产生列名，并创建序表。

B3：循环分组后的成员，将每组的数据动态拼在一起，添加到A3创建的序表中。

2.多表关联列转行

有三张表，分别是学生表、成绩表、补考成绩，以Stu_id为关联字段，如下图：



现在要查询三张表，得到每个学生的各科成绩、总成绩、补考成绩，如下图：

stu_id	stu_name	Chinese_score	Math_score	total_score	Chinese_retest	Math_retest
1	Ashley	80	77	156		
2	Rachel	58	67	125	78	
3	Emily	85	56	141		82

2.多表关联列转行

SPL脚本如下：

	A	B
1	=connect("db"). query("select t1.stu_id stu_id,t1.stu_name stu_name,t2.subject subject,t2.score score1,t3.score score2 from Students.txt t1 left join Exam.txt t2 on t1.stu_id=t2.stu_id left join Retest.txt t3 on t1.stu_id=t3.stu_id and t2.subject=t3.subject order by t1.stu_id,t2.subject")	
2	=A1.group(stu_id)	=A1.group(subject)
3	=create(stu_id,stu_name,${(B2.(~.subject+"_score")|"total_score"|B2.(~.subject+"_retest")).string()})	
4	>A2.run(A3.record([stu_id,stu_name]|B2.(~(A2.#).score1)|A2.sum(score1)|B2.(~(A2.#).score2)))	

A1：将订单表和订单明细表连接取数。A2：按订单ID分组。

B2~A3：根据分组内成员数量最多的组，动态产生列名，并创建序表。

B3：循环分组后的成员，将每组的数据动态拼在一起，添加到A3创建的序表中。

表间关联转置小结

表间关联转置先进行连接

首先数据表通过关联关系，连接成一张表。接下来就和前面介绍的转置类似了。

SPL的序表支持丰富的函数，可以胜任绝大部分的运算需求。

07

分栏

分栏

有一个世界城市人口表，如下图：

Continent	Country	City	Population
Africa	Egypt	Cairo	6789479
Asia	China	Shanghai	24240000
Europe	Britain	London	7285000

分栏列出欧洲和非洲人口超 200 万的城市名称及人口（每栏按从多到少排序），期望结果如下图：

Europe City	Population	Africa City	Population
Moscow	8389200	Cairo	6789479
London	7285000	Kinshasa	5064000
St Petersburg	4694000	Alexandria	3328196

分栏

分栏的思路也是先创建目标数据结构，再往里面填数据。

SPL脚本如下：

	A	B
1	=connect("db").query("select * from World where Continent in('Europe', 'Africa') and Population >= 2000000")	
2	=A1.select(Continent:"Europe")	=A1.select(Continent:"Africa")
3	=create('Europe City',Population,'Africa City', Population)	=A3.paste(A2.(City),A2.(Population),B2.(City),B2.(Population))

A1：连接数据库并取数，选出欧洲和非洲超过200万人口的记录。

A2~B2：分别取出欧洲和非洲的数据。

A3：按目标结构创建一个空序表。B3：使用序表的paste函数将值序列直接粘贴到对应列。

THANKS

感谢聆听 批评指导