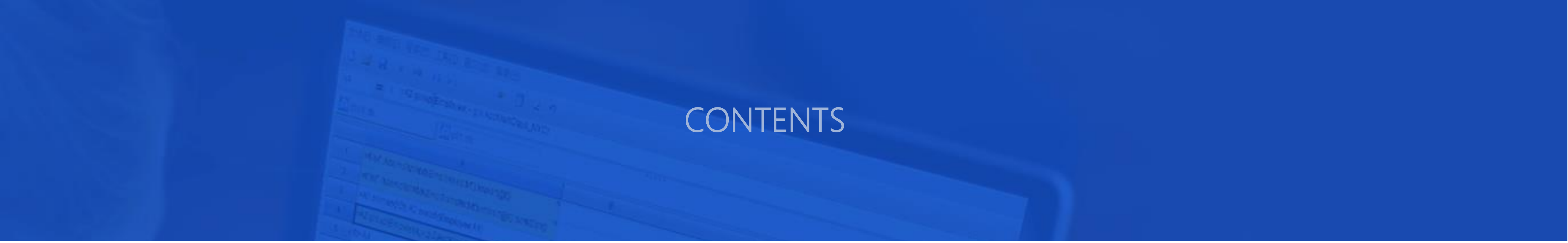
Grouped subsets

Raqsoft - esProc

www.raqsoft.com

Preface

The grouping of SQL will force the following aggregation operation, that is, the aggregation is performed when a large set is divided into subsets. But sometimes we are more concerned about the subsets after grouping, and do not want to aggregate immediately, so we need to split the grouping operation into two steps, so as to complete more complex calculation based on subsets.



CONTENTS

01

Equivalent grouping

02

Ordered grouping

03

Ordered conditional grouping

04

Sequence number grouping

05

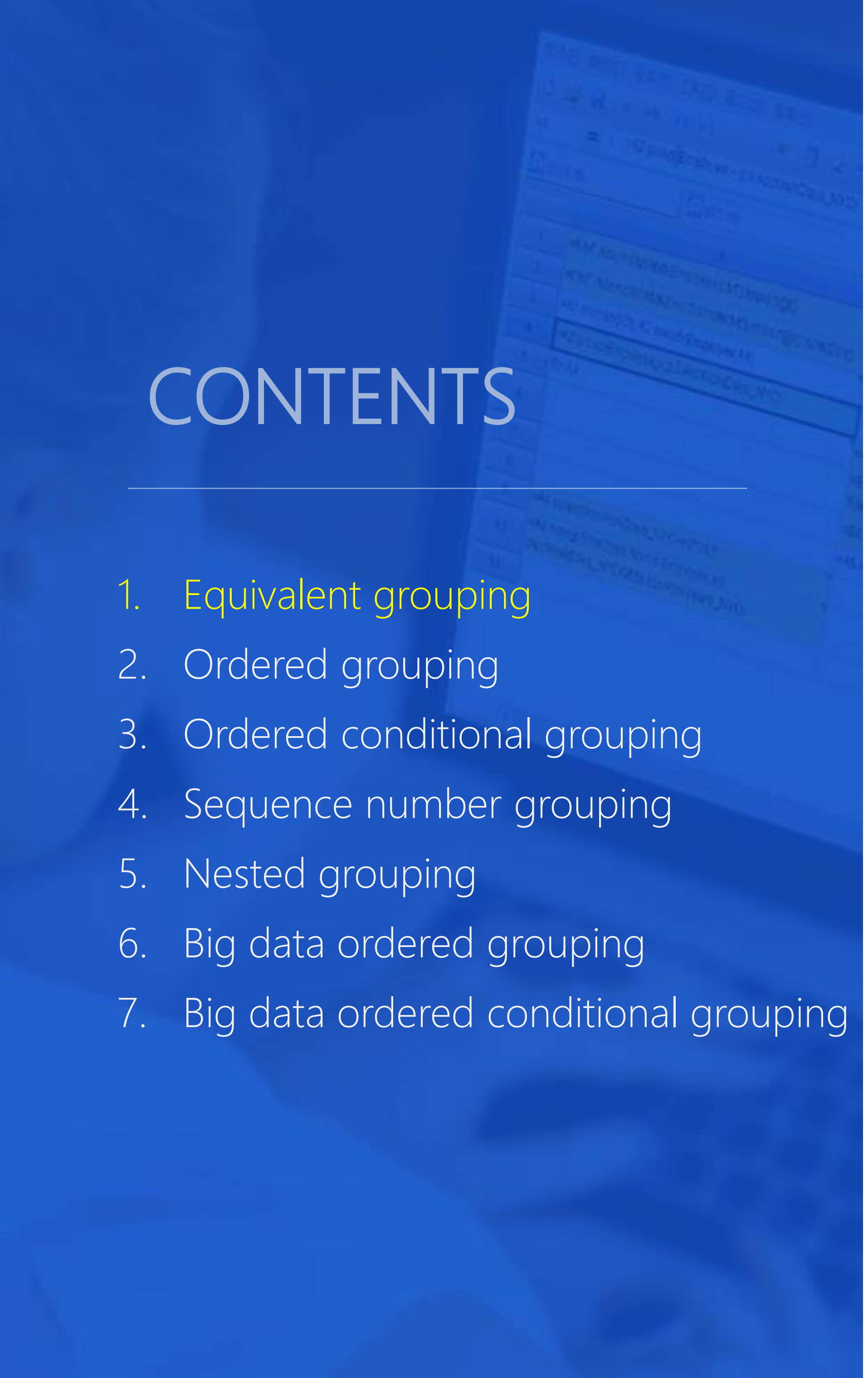
Nested grouping

06

Big data ordered grouping

07

Big data ordered conditional grouping



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

01

Equivalent grouping

Equivalent grouping

The so-called equivalent grouping, that is, the records with the same grouped field values are divided into a group, is the most common grouping method. The related functions are A.group@u() or A.group() , the former does not sort the grouping results, the latter will sort.

For example, there are data:

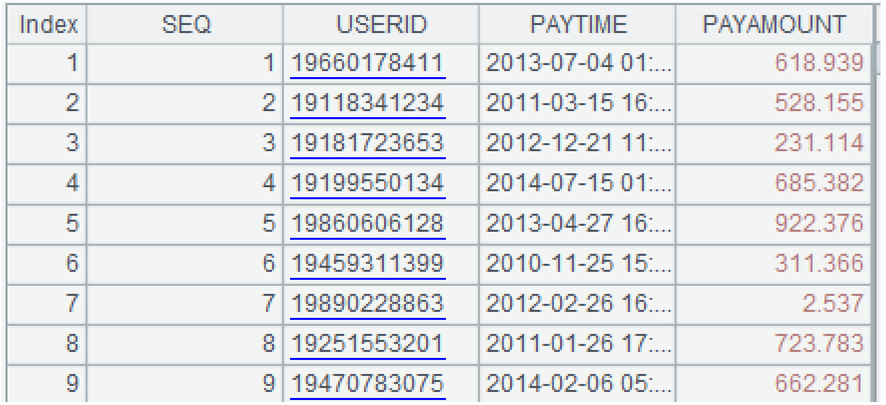
grp	calc
2	6.0
1	5.0
2	2.5
3	3.5
1	6.5

member
[{2,6.0},{2,2.5}]
[{1,5.0},{1,6.5}]
[{3,3.5}]

Equivalent grouping

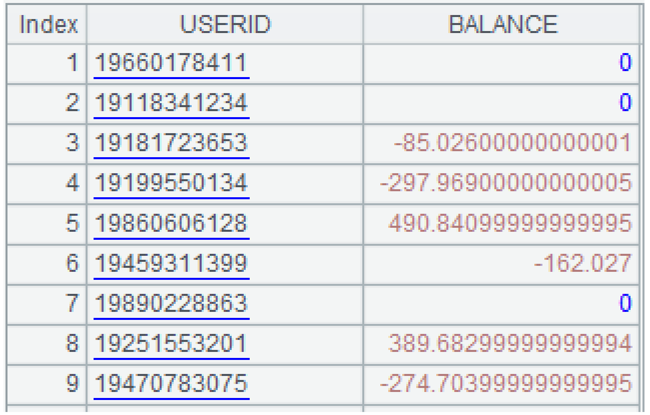
Example: there is a user consumption table. I want to know the difference between each user's last and previous consumption.

User consumption table



Index	SEQ	USERID	PAYTIME	PAYAMOUNT
1	1	19660178411	2013-07-04 01:...	618.939
2	2	19118341234	2011-03-15 16:...	528.155
3	3	19181723653	2012-12-21 11:...	231.114
4	4	19199550134	2014-07-15 01:...	685.382
5	5	19860606128	2013-04-27 16:...	922.376
6	6	19459311399	2010-11-25 15:...	311.366
7	7	19890228863	2012-02-26 16:...	2.537
8	8	19251553201	2011-01-26 17:...	723.783
9	9	19470783075	2014-02-06 05:...	662.281

SPL output



Index	USERID	BALANCE
1	19660178411	0
2	19118341234	0
3	19181723653	-85.02600000000001
4	19199550134	-297.96900000000005
5	19860606128	490.84099999999995
6	19459311399	-162.027
7	19890228863	0
8	19251553201	389.68299999999994
9	19470783075	-274.70399999999995

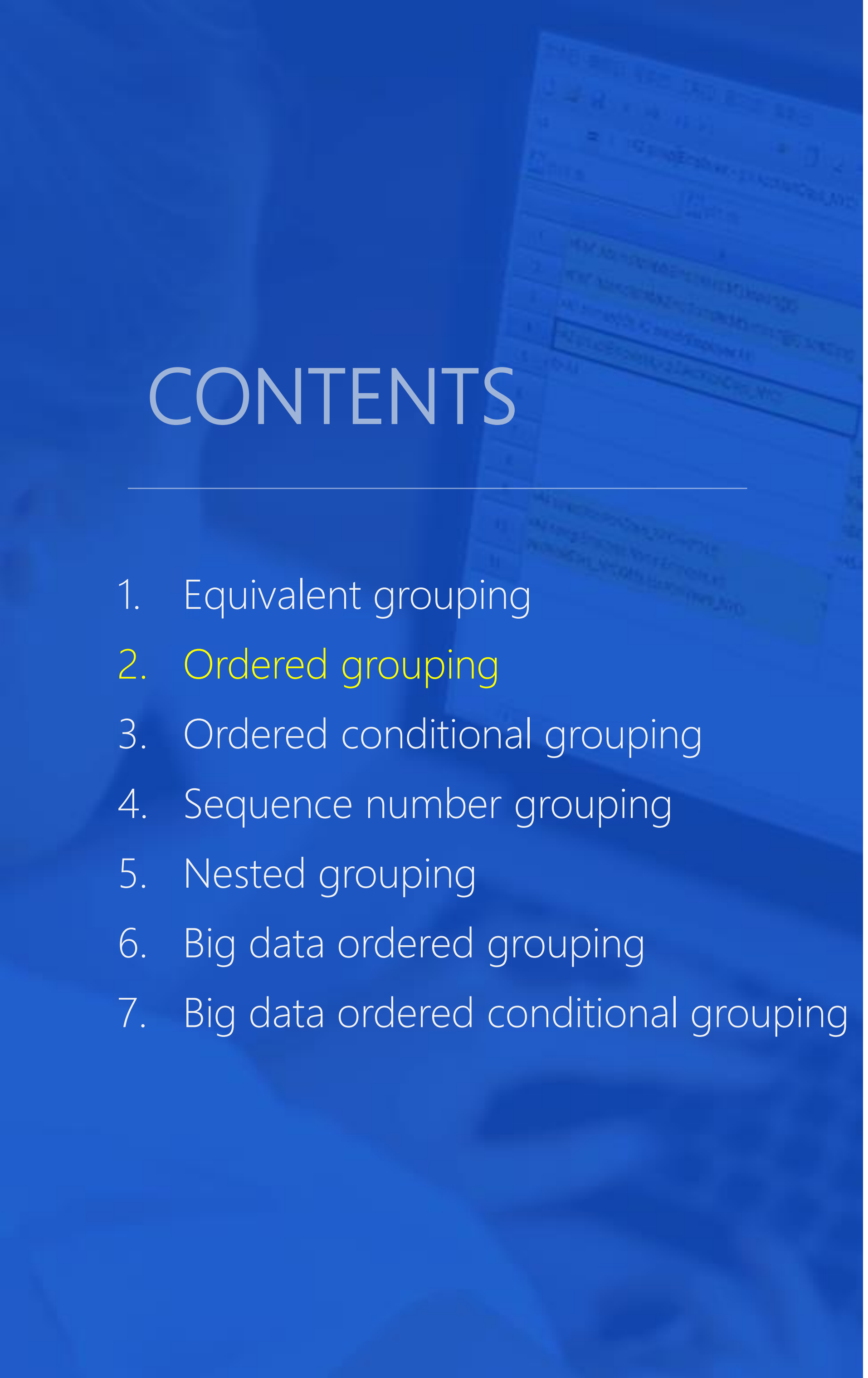
SPL code

	A
1	=db.query("SELECT * FROM USERPAY")
2	=A1.group@u(USERID)
3	=A2.(~.top(-2;PAYTIME))
4	=A3.new(~.USERID,if(~.count()<2,0,(~(1).PAYAMOUNT-~(2).PAYAMOUNT)):BALANCE)

A2: group by user, no sorting required

A3: Sort each group by consumption time and take the last two (if any)

A4: Calculate the difference between the last time and the previous time



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

02

Ordered grouping

Ordered grouping

If the grouping fields are already in order, the grouped subsets can be calculated during traversal. The related function is A.group@o().

Data 1:

grp	calc
1	5.0
1	6.5
2	6.0
2	2.5
3	3.5

Group by grp
during traversal

member
[{1,5.0},{1,6.5}]
[{2,6.0},{2,2.5}]
[{3,3.5}]

Data 2:

grp	calc
1	5.0
1	6.5
2	6.0
2	2.5
3	3.5
1	5.0

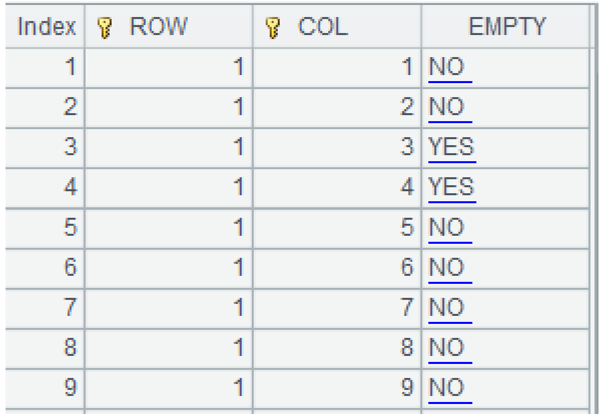
Group by grp
during traversal

member
[{1,5.0},{1,6.5}]
[{2,6.0},{2,2.5}]
[{3,3.5}]
[{1,5.0}]

Ordered grouping

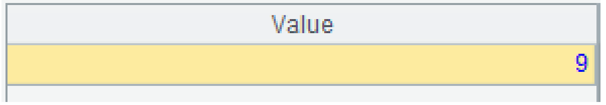
Example: in a movie theater, you want to count the maximum number of adjacent empty seats.

Empty seat table of a movie theater



Index	ROW	COL	EMPTY
1	1	1	NO
2	1	2	NO
3	1	3	YES
4	1	4	YES
5	1	5	NO
6	1	6	NO
7	1	7	NO
8	1	8	NO
9	1	9	NO

SPL output



Value
9

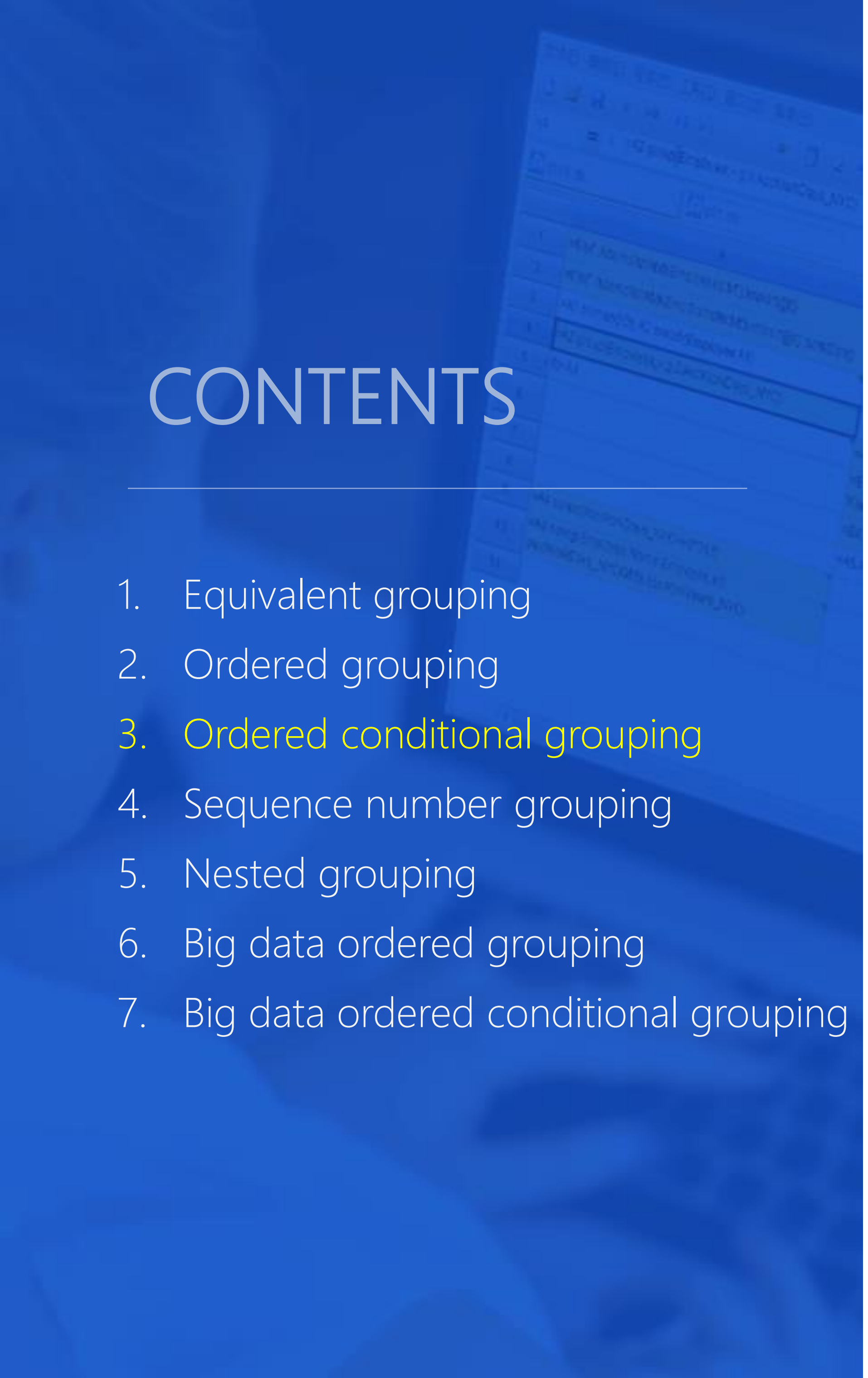
SPL code

	A
1	=file("D:/CINEMA.ctx").create().cursor().fetch()
2	=A1.group@o(ROW,EMPTY;~.count():CNT)
3	=A2.select(EMPTY=="YES")
4	=A3.max(CNT)

A2: Group data in original order

A3: Filter out the non empty seat

A4: Find the maximum adjacent empty seats



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

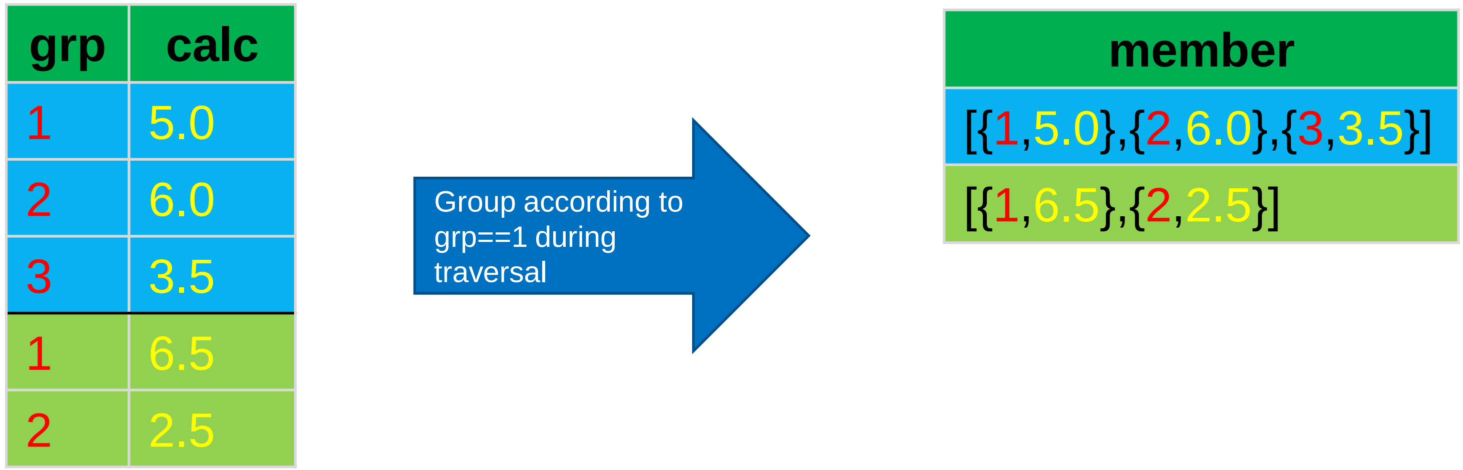
03

Ordered conditional grouping

Ordered conditional grouping

Ordered grouping is equivalent to dividing a new group in an ordered sequence whenever the value of the grouping field changes. But sometimes it is not the field value that changes, but other conditions, which need to be handled by the function of ordered conditional grouping. The related function is: A.group@i()

For example, there are data:



grp	calc
1	5.0
2	6.0
3	3.5
1	6.5
2	2.5

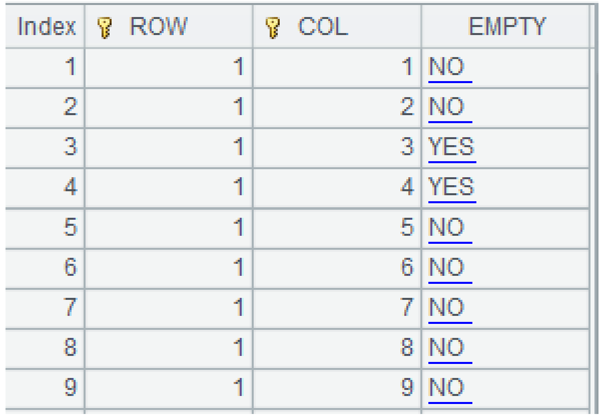
Group according to grp==1 during traversal

member
[{1,5.0},{2,6.0},{3,3.5}]
[{1,6.5},{2,2.5}]

Ordered conditional grouping

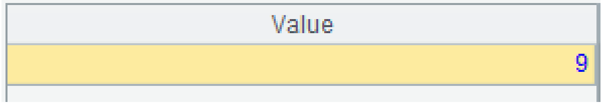
Example: for the problem of adjacent empty seats in the previous chapter, ordered conditional grouping can also be used.

Empty seat table of a movie theater



Index	ROW	COL	EMPTY
1	1	1	NO
2	1	2	NO
3	1	3	YES
4	1	4	YES
5	1	5	NO
6	1	6	NO
7	1	7	NO
8	1	8	NO
9	1	9	NO

SPL output



Value
9

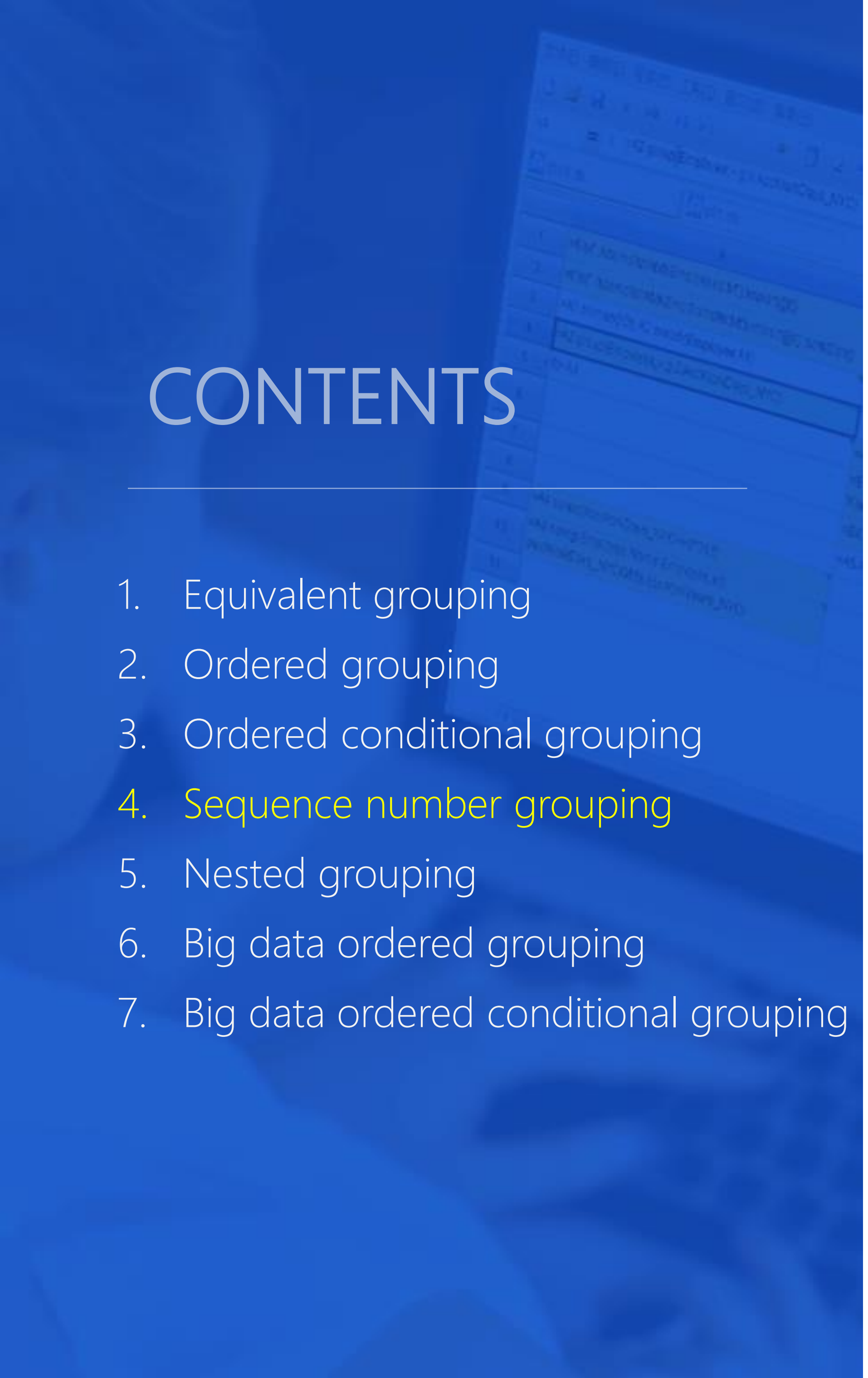
SPL code

	A
1	=file("D:/CINEMA.ctx").create().cursor().fetch()
2	=A1.group@i(ROW[-1]!=ROW||EMPTY[-1]!=EMPTY;EMPTY,~.count():CNT)
3	=A2.select(EMPTY=="YES")
4	=A3.max(CNT)

A2: Conditional grouping of data in the original order

A3: Filter out the non empty seat

A4: Find the maximum adjacent empty seats



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

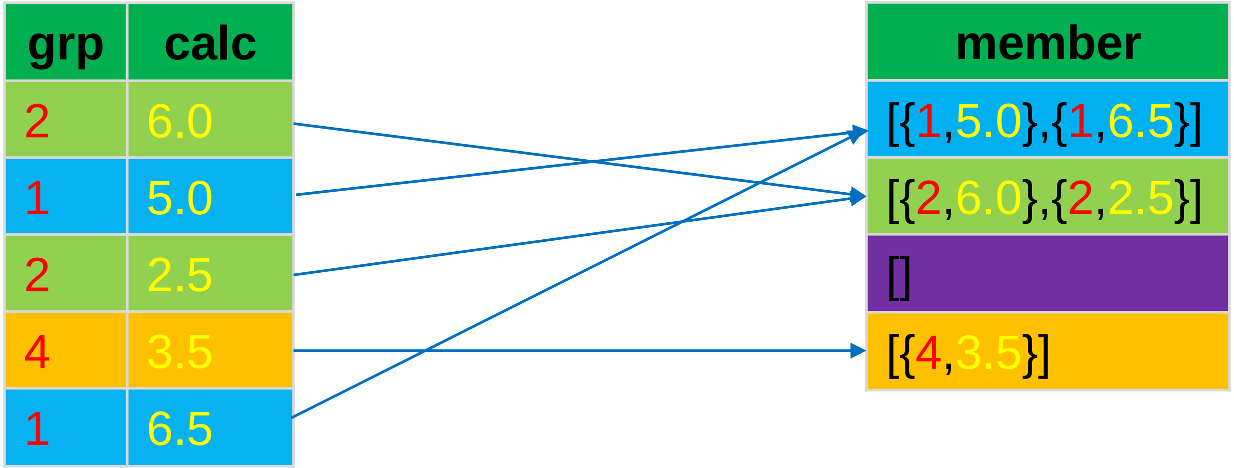
04

Sequence number grouping

Sequence number grouping

Sequence number grouping uses the inherent sequence of sequence number for direct positioning (grouping is actually built in advance), so the result of sequence number grouping is orderly in nature. It can also be used for sorting. The related function is: A.group@n()

For example, there are data:



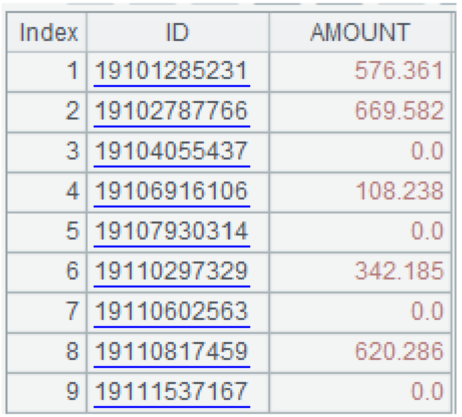
grp	calc
2	6.0
1	5.0
2	2.5
4	3.5
1	6.5

member
[{1,5.0},{1,6.5}]
[{2,6.0},{2,2.5}]
[]
[{4,3.5}]

Sequence number grouping

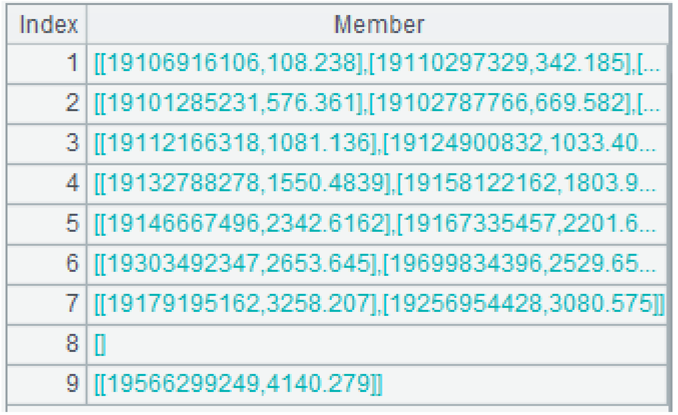
Example： There is an account balance table. If you remove the balance of 0, the rest will be grouped into groups for each difference of 500.

Account balance table



Index	ID	AMOUNT
1	19101285231	576.361
2	19102787766	669.582
3	19104055437	0.0
4	19106916106	108.238
5	19107930314	0.0
6	19110297329	342.185
7	19110602563	0.0
8	19110817459	620.286
9	19111537167	0.0

SPL output



Index	Member
1	[[19106916106,108.238],[19110297329,342.185],[...
2	[[19101285231,576.361],[19102787766,669.582],[...
3	[[19112166318,1081.136],[19124900832,1033.40...
4	[[19132788278,1550.4839],[19158122162,1803.9...
5	[[19146667496,2342.6162],[19167335457,2201.6...
6	[[19303492347,2653.645],[19699834396,2529.65...
7	[[19179195162,3258.207],[19256954428,3080.575]]
8	[]
9	[[19566299249,4140.279]]

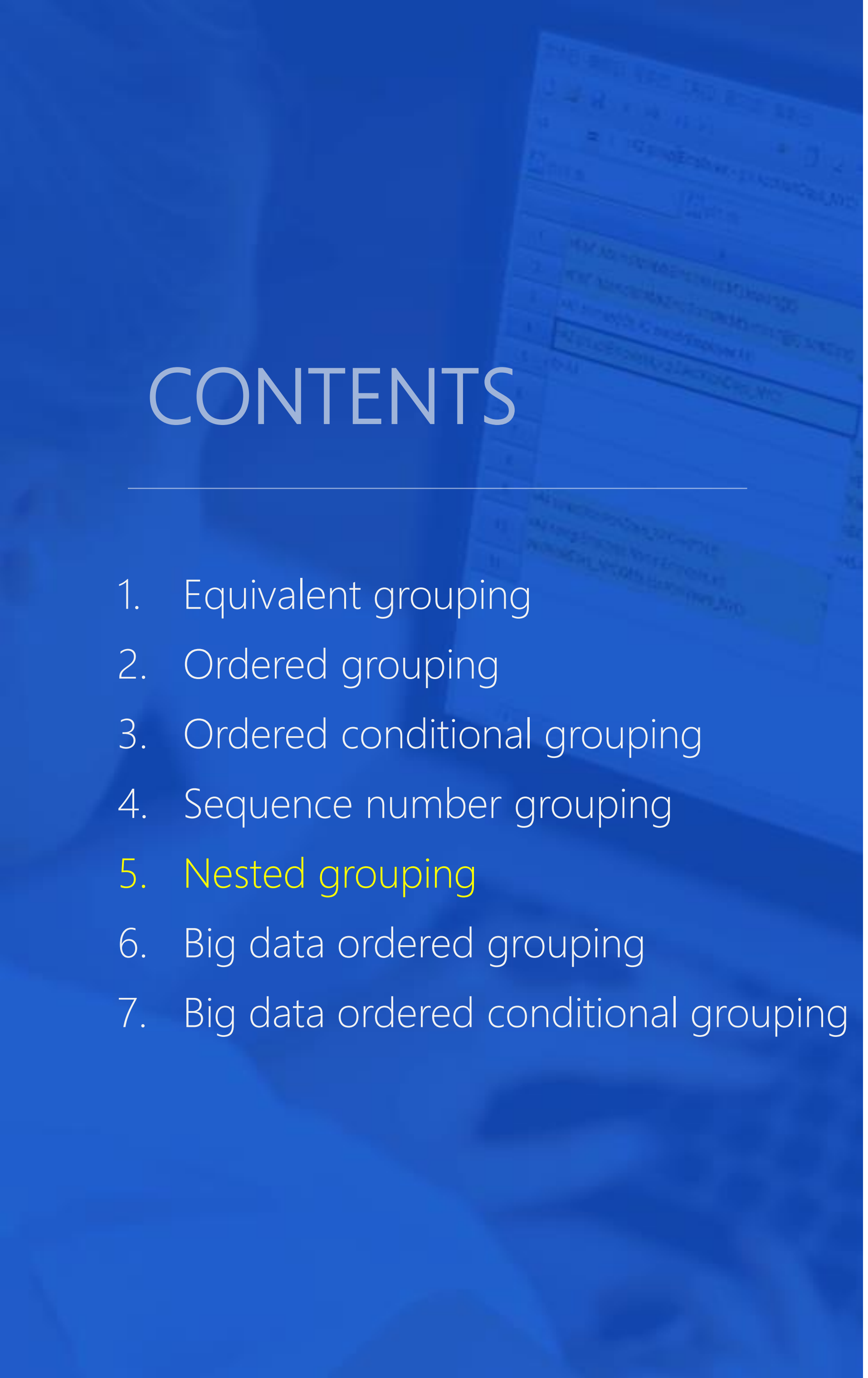
You can see that there are groups with empty members in the SPL results, because sequence number grouping directly locates by sequence number. So you can quickly find the required subsets: for example, the parts of 3000-3500 should be in group 7.

SPL code

	A
1	=db.query("SELECT * FROM USERACCOUNT WHERE AMOUNT > 0.000001")
2	=A1.group@n(int(AMOUNT/500)+1)

A1: Remove balance 0 from data

A2: Divide the data into groups of 500 yuan



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

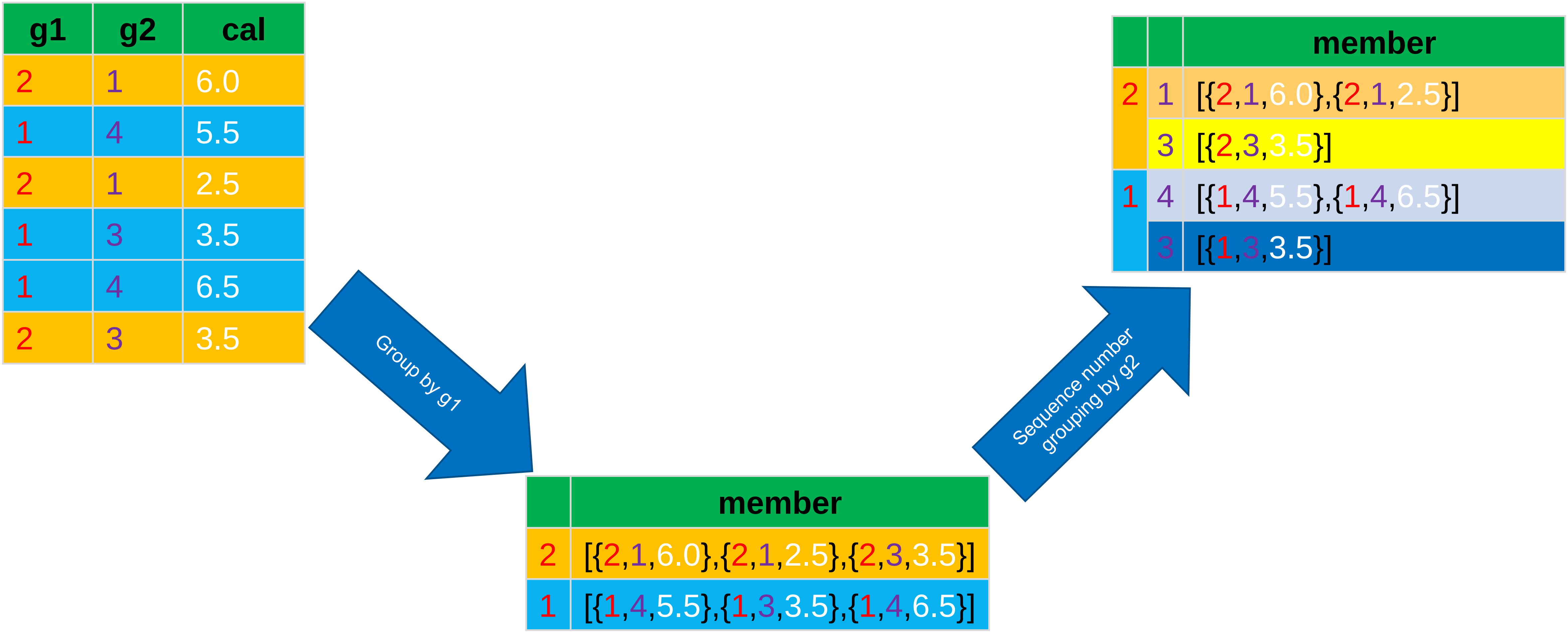
05

Nested grouping

Nested grouping

Nested grouping, that is, the original data used in one grouping calculation, is the result of another grouping calculation.

For example：



g1	g2	cal
2	1	6.0
1	4	5.5
2	1	2.5
1	3	3.5
1	4	6.5
2	3	3.5

Group by g1

	member
2	[{2,1,6.0},{2,1,2.5},{2,3,3.5}]
1	[{1,4,5.5},{1,3,3.5},{1,4,6.5}]

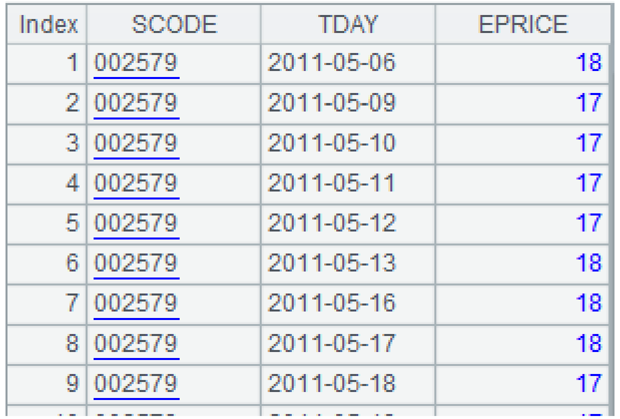
Sequence number grouping by g2

		member
2	1	[{2,1,6.0},{2,1,2.5}]
	3	[{2,3,3.5}]
1	4	[{1,4,5.5},{1,4,6.5}]
	3	[{1,3,3.5}]

Nested grouping

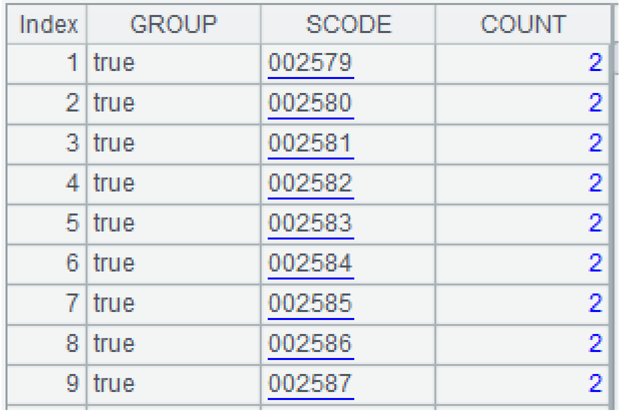
Example: There is a daily trading summary of the stock market. I want to know the maximum number of consecutive days for the closing price of each stock to rise.

Stock market daily trading summary table



Index	SCODE	TDAY	EPRICE
1	002579	2011-05-06	18
2	002579	2011-05-09	17
3	002579	2011-05-10	17
4	002579	2011-05-11	17
5	002579	2011-05-12	17
6	002579	2011-05-13	18
7	002579	2011-05-16	18
8	002579	2011-05-17	18
9	002579	2011-05-18	17

SPL output



Index	GROUP	SCODE	COUNT
1	true	002579	2
2	true	002580	2
3	true	002581	2
4	true	002582	2
5	true	002583	2
6	true	002584	2
7	true	002585	2
8	true	002586	2
9	true	002587	2

SPL code

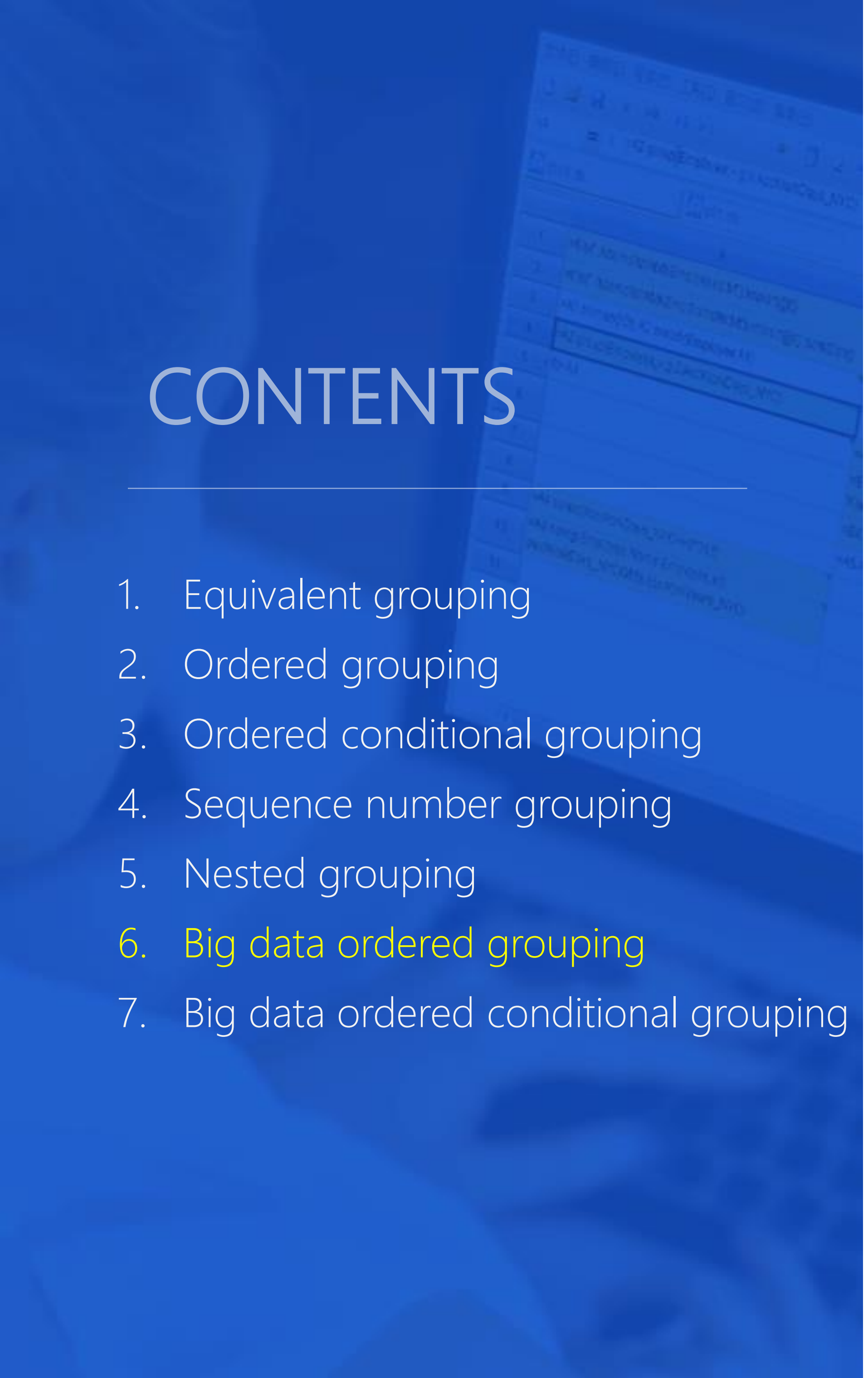
	A
1	=file("D:/STOCK.ctx").create().cursor().fetch()
2	=A1.group@u(SCODE)
3	=A2.(~.sort(TDAY))
4	=A3.(~.group@i((EPRICE<=EPRICE[-1]):GROUP;SCODE,(~.count()-1):COUNT))
5	=A4.(~.maxp(COUNT))

A2: Group data by stock code

A3: Sort data within a group by date

A4: Regroup and count according to whether it is rising continuously or not

A5: Take the most days in each group



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

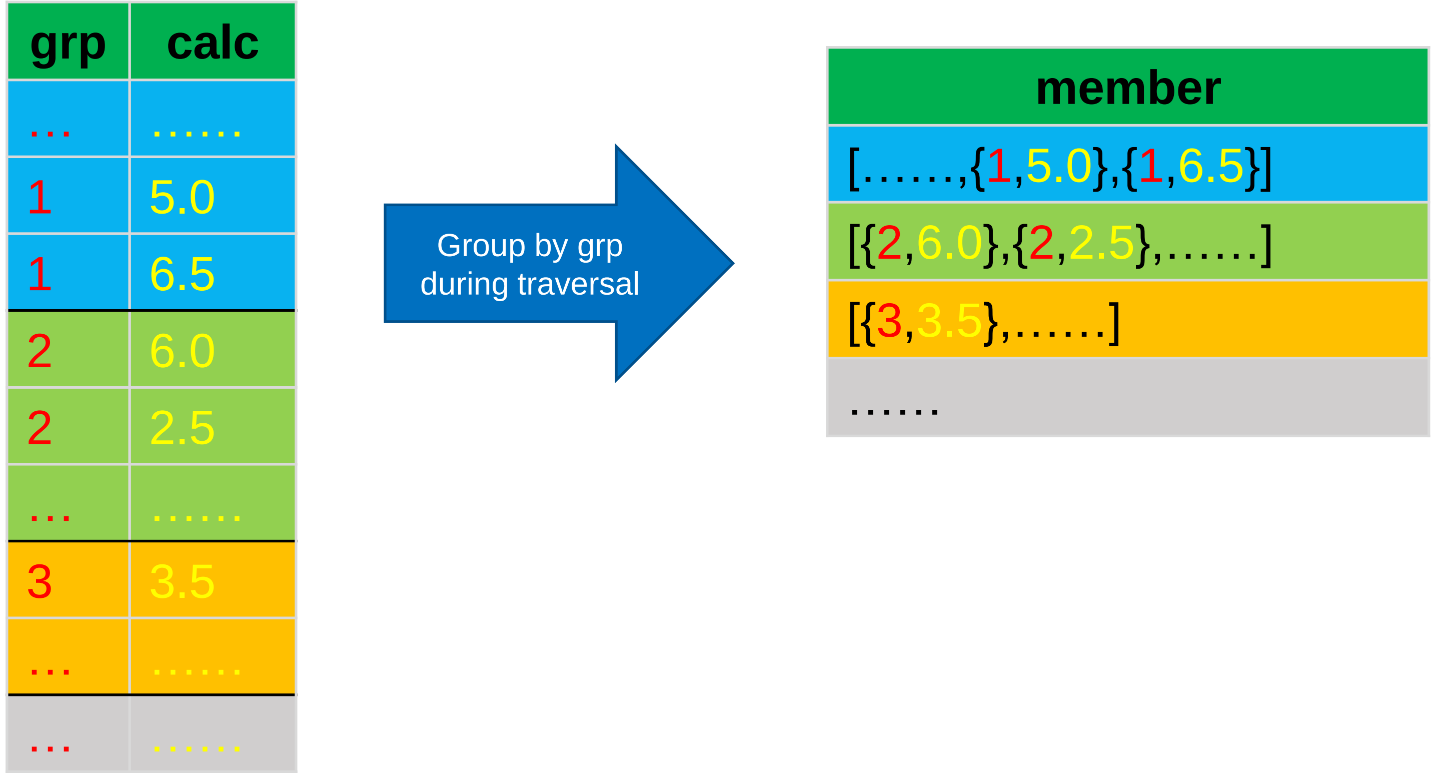
06

Big data ordered grouping

Big data ordered grouping

Big data, that is, a large amount of data that memory can't hold, so it's impossible to save all the grouped subsets. In order to read and process the data at the same time, it needs to be ordered, because the ordered data can be regarded as having done the grouping in advance, only to find the boundary of the grouping. The related function is: cs.group()

For example, there are data:



grp	calc
…	……
1	5.0
1	6.5
2	6.0
2	2.5
…	……
3	3.5
…	……
…	……

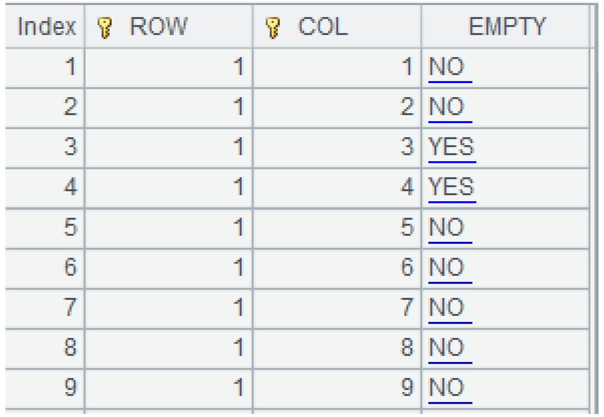
Group by grp during traversal

member
[……,{1,5.0},{1,6.5}]
[{2,6.0},{2,2.5},……]
[{3,3.5},……]
……

Big data ordered grouping

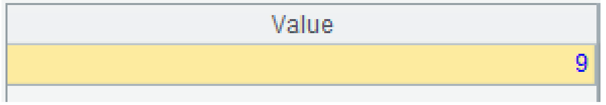
Example： for the problem of adjacent empty seats of ordered grouping in the previous chapter, in the case of big data:

Empty seat table of a movie theater



Index	ROW	COL	EMPTY
1	1	1	NO
2	1	2	NO
3	1	3	YES
4	1	4	YES
5	1	5	NO
6	1	6	NO
7	1	7	NO
8	1	8	NO
9	1	9	NO

SPL output



Value
9

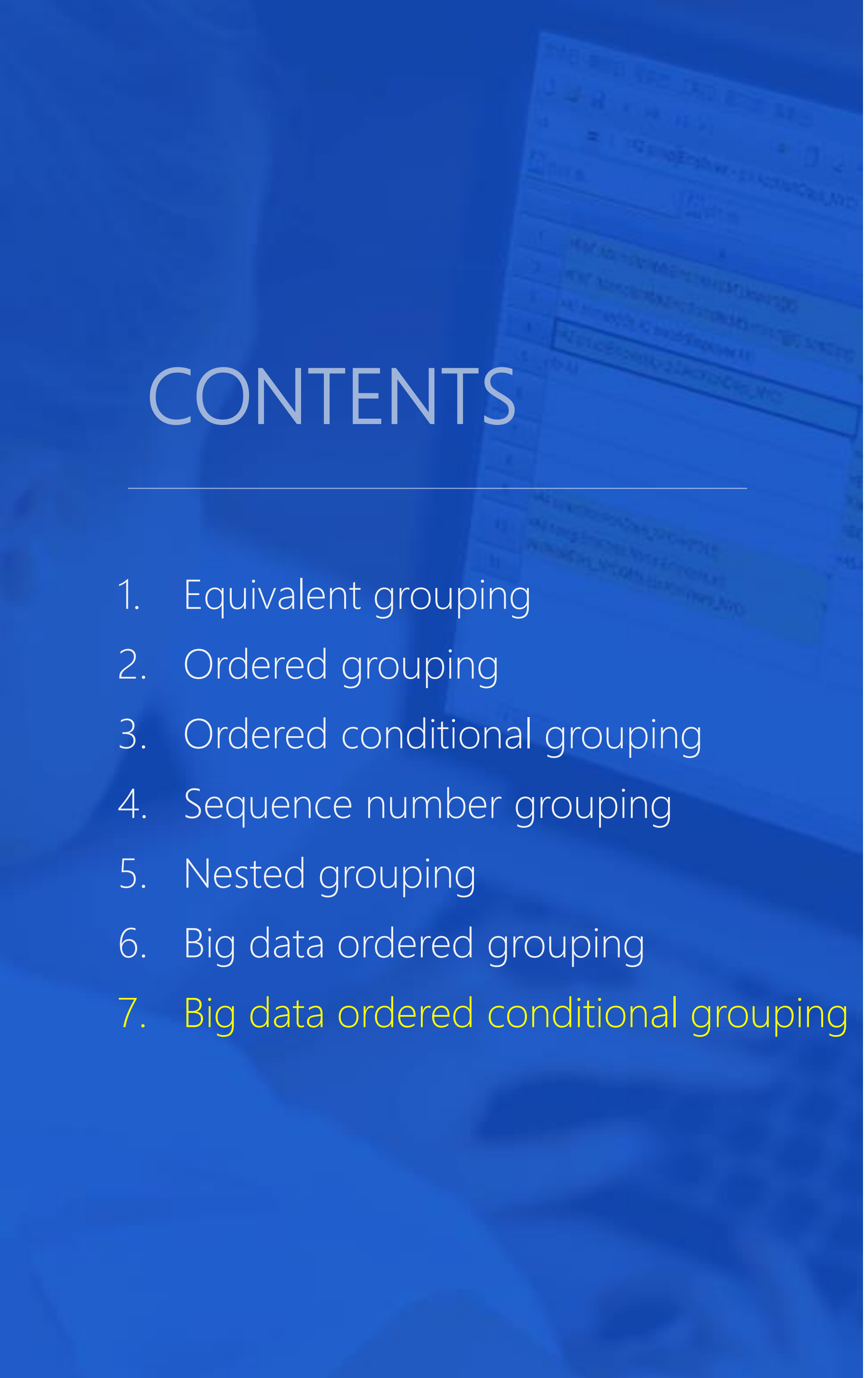
SPL code

	A
1	=file("D:/CINEMA.ctx").create().cursor()
2	=A1.group(ROW,EMPTY;count(1):CNT)
3	=A2.select(EMPTY=="YES")
4	=A3.groups(;max(CNT):CNT)(1).CNT

A2: Group data in original order

A3: Filter out the non empty seat

A4: Find the maximum adjacent empty seats



CONTENTS

1. Equivalent grouping
2. Ordered grouping
3. Ordered conditional grouping
4. Sequence number grouping
5. Nested grouping
6. Big data ordered grouping
7. Big data ordered conditional grouping

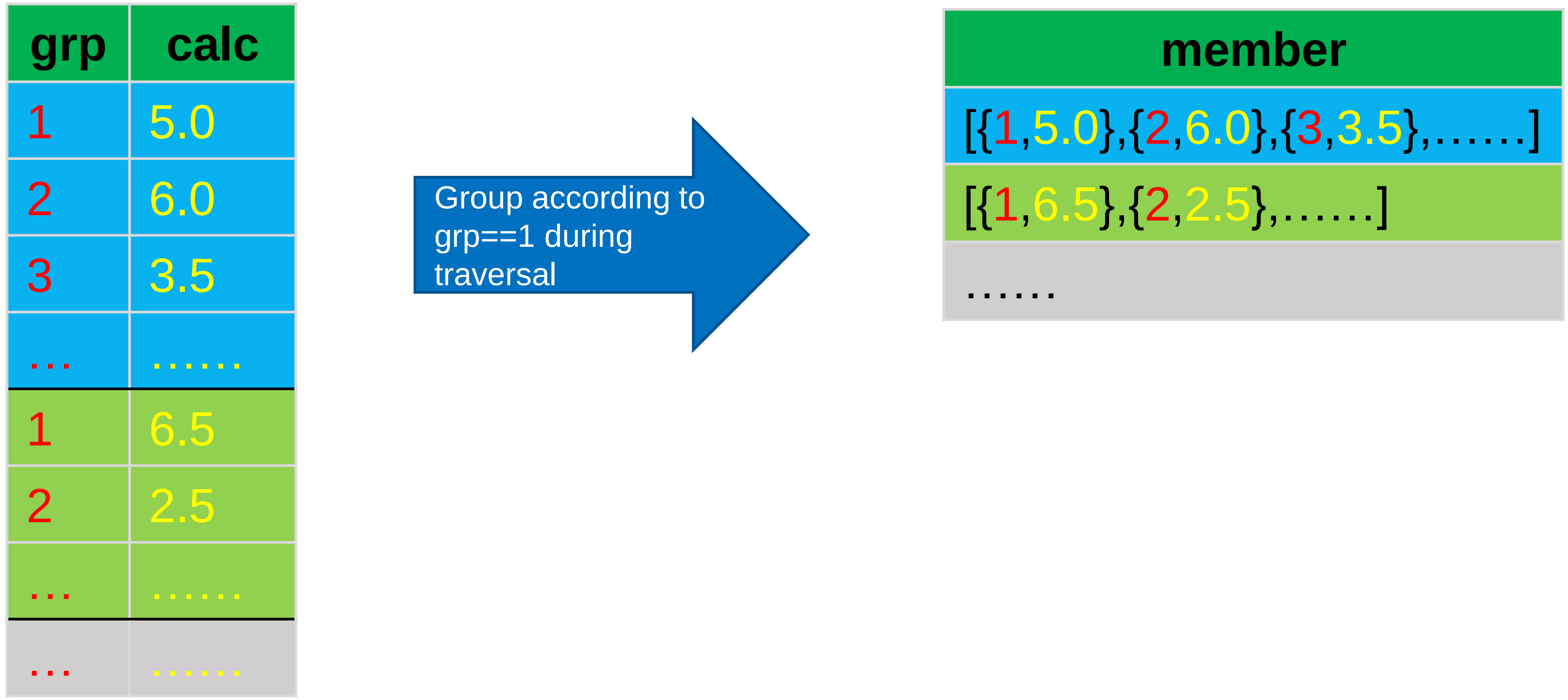
07

Big data ordered conditional grouping

Big data ordered conditional grouping

Just as ordered grouping can be regarded as a special case of ordered conditional grouping, ordered grouping of big data can also be regarded as a special case of ordered conditional grouping of big data. The related function is: cs.group@i()

For example, there are data:



grp	calc
1	5.0
2	6.0
3	3.5
…	……
1	6.5
2	2.5
…	……
…	……

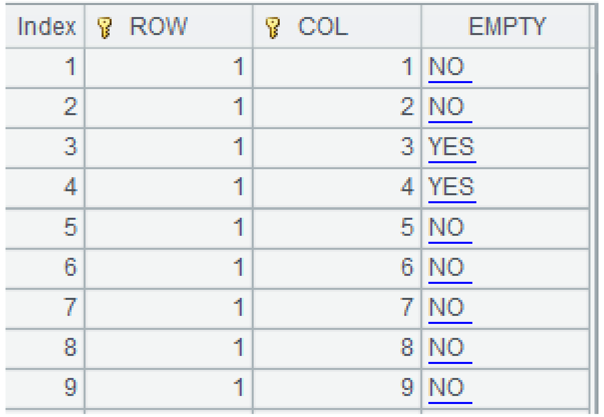
Group according to grp==1 during traversal

member
[{1,5.0},{2,6.0},{3,3.5},……]
[{1,6.5},{2,2.5},……]
……

Big data ordered conditional grouping

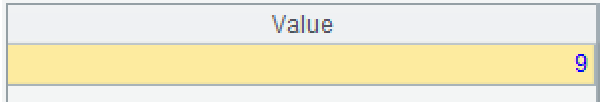
Example： for the problem of adjacent empty seats of ordered grouping in the previous chapter, in the case of big data:

Empty seat table of a movie theater



Index	ROW	COL	EMPTY
1	1	1	NO
2	1	2	NO
3	1	3	YES
4	1	4	YES
5	1	5	NO
6	1	6	NO
7	1	7	NO
8	1	8	NO
9	1	9	NO

SPL output



Value
9

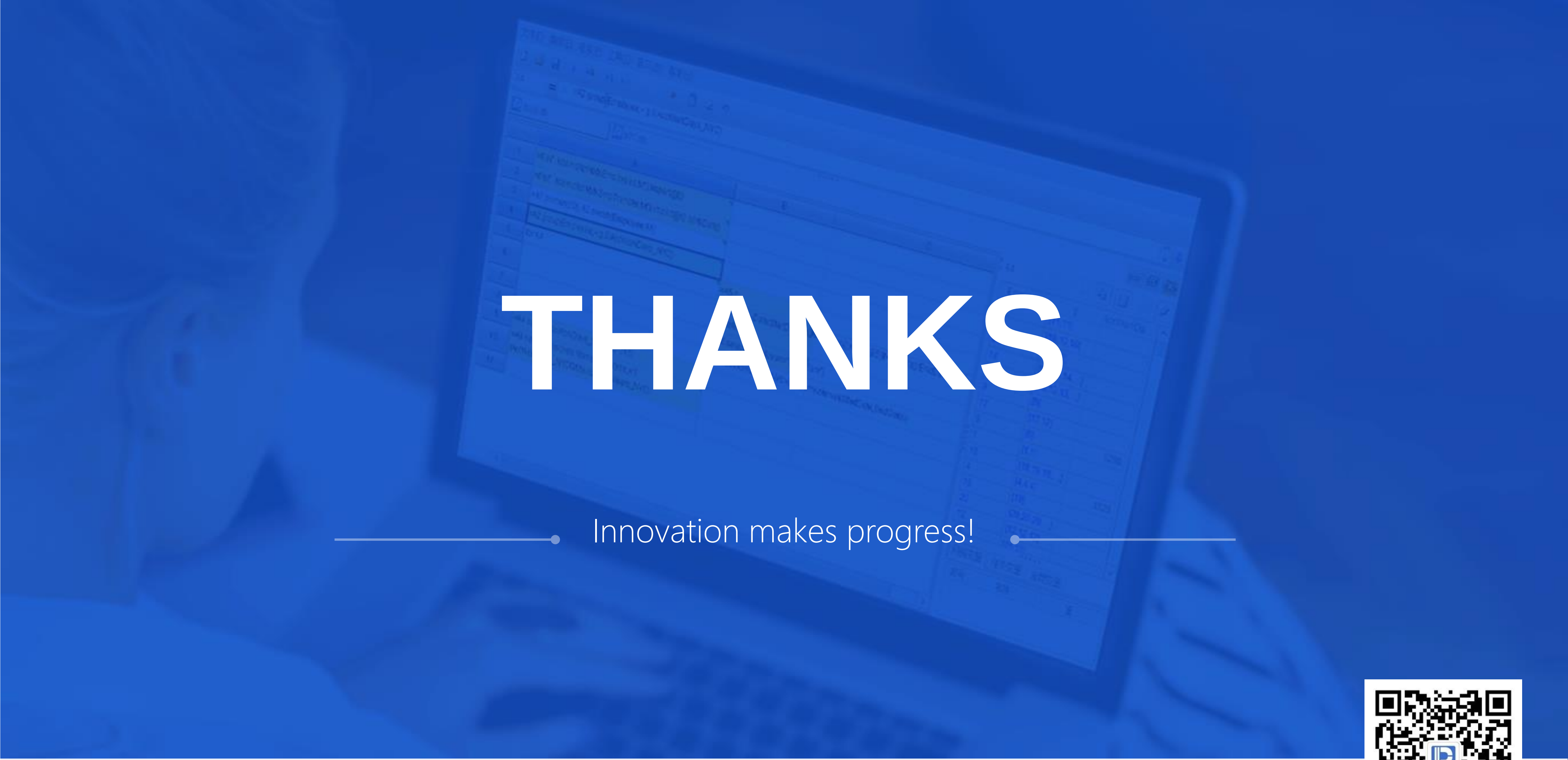
SPL code

	A
1	=file("D:/CINEMA.ctx").create().cursor()
2	=A1.group@i(ROW[-1]!=ROW||EMPTY[-1]!=EMPTY;EMPTY,count(1):CNT)
3	=A2.select(EMPTY=="YES")
4	=A3.groups(;max(CNT):CNT)(1).CNT

A2: Conditional grouping of data in the original order

A3: Filter out the non empty seat

A4: Find the maximum adjacent empty seats



THANKS

Innovation makes progress!

www.raqsoft.com