Grouping

CONTENTS

01

Grouped subsets

Example of regular grouping: counting the number of employees in each department

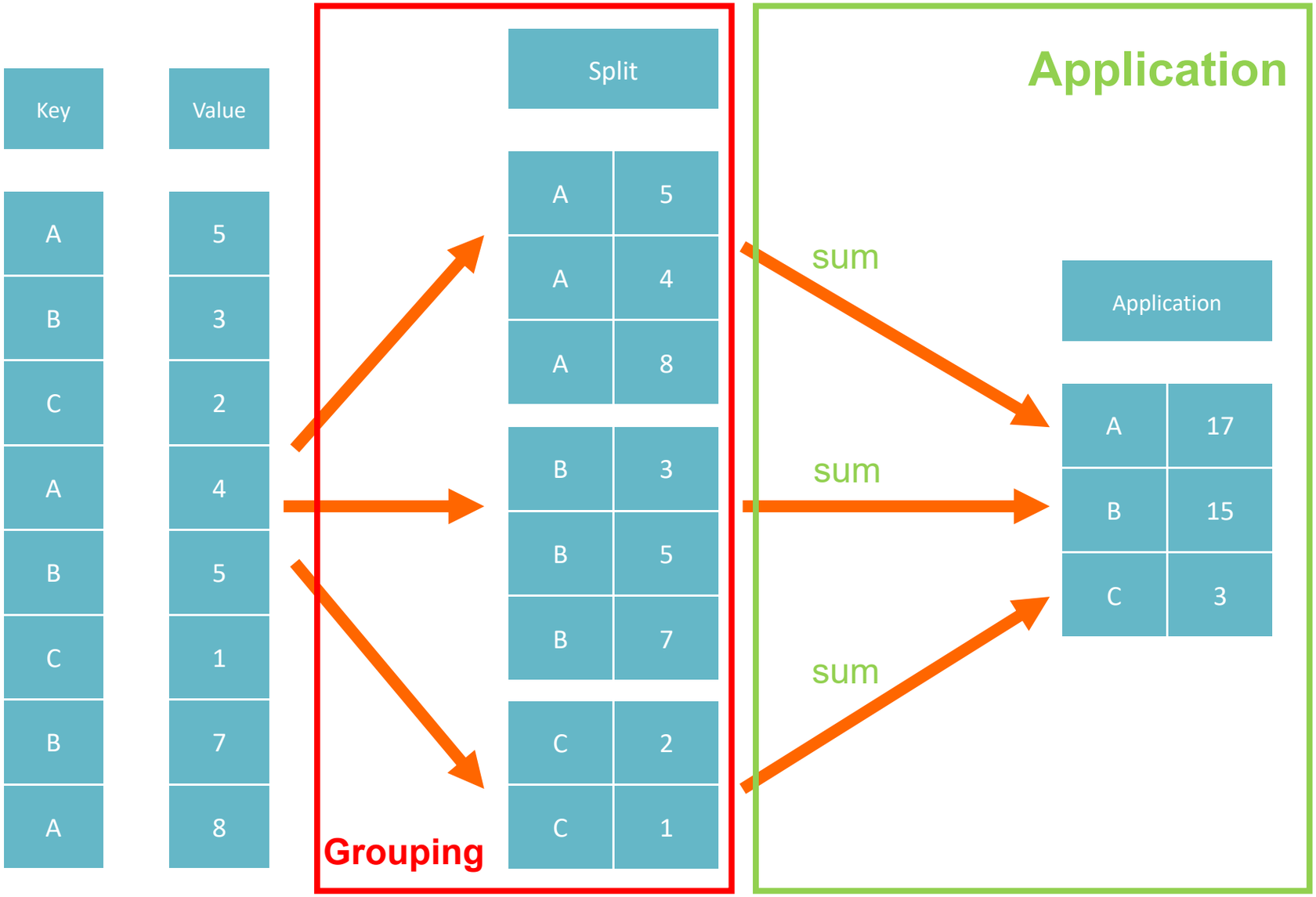
	A
1	=demo.query("select DEPT,count(*) from EMPLOYEE group by DEPT")
2	=demo.query("select * from EMPLOYEE").groups(DEPT;count(~):num)

A1,A2 results

Index	DEPT	C2
1	R&D	29
2	Finance	24
3	Sales	187
4	HR	19
5	Marketing	99
6	Production	91
7	Administration	4
8	Technology	47

Index	DEPT	num
1	Administration	4
2	Finance	24
3	HR	19
4	Marketing	99
5	Production	91
6	R&D	29
7	Sales	187
8	Technology	47

The Nature of Grouping



SQL grouping forces aggregation. But sometimes we are interested not only in aggregated values, but also in grouped subsets.

Example: Find out who has the same birthday as other employees.

	A	B
1	=demo.query("select NAME,GENDER,BIRTHDAY from EMPLOYEE")	
2	=A1.group(BIRTHDAY)	/grouped subsets，***no aggregation***
3	=A2.select(~.len()>2).conj()	/Select the subsets with number greater than 2

A1

Index	NAME	GENDER	BIRTHDAY
1	Rebecca	F	1974-11-20
2	Ashley	F	1980-07-19
3	Rachel	F	1970-12-17
4	Emily	F	1985-03-07
5	Ashley	F	1975-05-13
6	Matthew	M	1984-07-07
7	Alexis	F	1972-08-16
8	Megan	F	1979-04-19
9	Victoria	F	1983-12-07
10	Ryan	M	1976-03-12

A2

Index	Member
1	[[Olivia,F,1968-11-05]]
2	[[Alexis,F,1968-11-12]]
3	[[Nicholas,M,1968-11-24]]
4	[[Luis,M,1968-11-27]]
5	[[Tyler,M,1968-12-23]]
6	[[Sarah,F,1968-12-29]]
7	[[Jacob,M,1969-01-08]]
8	[[Jessica,F,1969-01-10]]
9	[[Ryan,M,1969-01-11]]
10	[[Michael,M,1969-01-18]]

Grouped subsets

Index	NAME	GENDER	BIRTHDAY
1	Olivia	F	1968-11-05

A3

Index	NAME	GENDER	BIRTHDAY
1	Andrew	M	1971-08-27
2	Elizabeth	F	1971-08-27
3	Olivia	F	1971-08-27
4	Alyssa	F	1977-12-24
5	Timothy	M	1977-12-24
6	Emily	F	1977-12-24
7	Jessica	F	1977-12-24
8	David	M	1977-12-24
9	Chloe	F	1978-08-20
10	Justin	M	1978-08-20

Check the scores of students whose total score is not less than 240

	A	B
1	=file("E:/txt/student_scores.txt").import@t()	
2	=A1.group(STUDENTID)	/grouped subsets，***no aggregation***
3	=A2.select(~.sum(SCORE)>=240).conj()	/Select students not less than 240 and merge them

A1

Index	STUDENTID	SUBJECT	SCORE
1	1	English	84
2	1	Math	77
3	1	PE	69
4	2	English	81
5	2	Math	80
6	2	PE	97
7	3	English	75
8	3	Math	86
9	3	PE	67
10	4	English	96

A2

Index	Member
1	[[1,English,84],[1,Math,77],[1,PE,69]]
2	[[2,English,81],[2,Math,80],[2,PE,97]]
3	[[3,English,75],[3,Math,86],[3,PE,67]]
4	[[4,English,96],[4,Math,63],[4,PE,81]]
5	[[5,English,72],[5,Math,60],[5,PE,91]]

Grouped subsets

Index	STUDENTID	SUBJECT	SCORE
1	1	English	84
2	1	Math	77
3	1	PE	69

A3

Index	STUDENTID	SUBJECT	SCORE
1	2	English	81
2	2	Math	80
3	2	PE	97
4	4	English	96
5	4	Math	63
6	4	PE	81

Grouping itself is a complex operation, which can be reused to improve the efficiency of operation by retaining the subset of grouping.

Example: Calculate the number of employees in each department, and then calculate the average age of staff in departments with more than 50 people.

	A	B
1	=demo.query("select NAME,GENDER,BIRTHDAY,DEPT from EMPLOYEE")	
2	=A1.group(DEPT)	/grouped subsets，***no aggregation***
3	=A2.new(DEPT,count(~):number)	/Calculate the number of employees in departments
4	=A2.select(~.count()>50).new(DEPT,int(~.avg(age(BIRTHDAY))):AGE)	/Calculate the average age of staff in departments with more than 50 people.

A1

Index	NAME	GENDER	BIRTHDAY	DEPT
1	Rebecca	F	1974-11-20	R&D
2	Ashley	F	1980-07-19	Finance
3	Rachel	F	1970-12-17	Sales
4	Emily	F	1985-03-07	HR
5	Ashley	F	1975-05-13	R&D
6	Matthew	M	1984-07-07	Sales
7	Alexis	F	1972-08-16	Sales
8	Megan	F	1979-04-19	Marketing
9	Victoria	F	1983-12-07	HR
10	Ryan	M	1976-03-12	R&D

A2

Index	Member
1	[[Jonathan,M,1971-03-07, ...],[Alexis,F,1977-08-07, ...],...
2	[[Ashley,F,1980-07-19, ...],[Daniel,M,1982-05-14, ...],[J...
3	[[Emily,F,1985-03-07, ...],[Victoria,F,1983-12-07, ...],[M...
4	[[Megan,F,1979-04-19, ...],[Hannah,F,1980-07-19, ...],[...
5	[[Christopher,M,1979-06-27, ...],[Samantha,F,1974-03...
6	[[Rebecca,F,19
7	[[Rachel,F,1970
8	[[Olivia,F,1971-

Grouped subsets

Index	NAME	GENDER	BIRTHDAY	DEPT
1	Jonathan	M	1971-03-07	Administration
2	Alexis	F	1977-08-07	Administration
3	Timothy	M	1977-12-24	Administration
4	Michael	M	1978-08-20	Administration

A3

Index	DEPT	number
1	Administrat...	4
2	Finance	24
3	HR	19
4	Marketing	99
5	Production	91
6	R&D	29
7	Sales	187
8	Technology	47

A4

Index	DEPT	AGE
1	Marketing	41
2	Production	40
3	Sales	40

Grouping subset is actually a set of sets. The set of sets can continue to be grouped into a set of sets (three-level set).

Example: after grouping by year, continue grouping by month

	A	B
1	=demo.query("select NAME,GENDER,BIRTHDAY,DEPT from EMPLOYEE")	
2	=A1.group(year(BIRTHDAY))	/Group by year
3	=A2.(~.group(month(BIRTHDAY)))	/Group by month after group by year

A1

Index	NAME	GENDER	BIRTHDAY	DEPT
1	Rebecca	F	1974-11-20	R&D
2	Ashley	F	1980-07-19	Finance
3	Rachel	F	1970-12-17	Sales
4	Emily	F	1985-03-07	HR
5	Ashley	F	1975-05-13	R&D
6	Matthew	M	1984-07-07	Sales
7	Alexis	F	1972-08-16	Sales
8	Megan	F	1979-04-19	Marketing
9	Victoria	F	1983-12-07	HR
10	Ryan	M	1976-03-12	R&D

A2

Index	Member
1	[[Sarah,F,1968-12-29, ...],[Luis,M,1968-11-27, ...],[Olivia,F,...
2	[[Kayla,F,1969-07-06, ...],[David,M,1969-01-31, ...],[Willia...
3	[[Rachel,F,1970-12-17, ...],[Cameron,M,1970-03-14, ...],[G...
4	[[Jonathan,M,1971-03-0 ...],[Andrew,M,1971-08-27, ...],[...
5	[[Alexis,F,1972-08-16, ...],[Matthew,M,1972-11-20, ...],[Tyle...

Index	NAME	GENDER	BIRTHDAY	DEPT
1	Sarah	F	1968-12-29	Marketing
2	Luis	M	1968-11-27	Technology
3	Olivia	F	1968-11-05	Production
4	Tyler	M	1968-12-23	Sales
5	Nicholas	M	1968-11-24	Marketing
6	Alexis	F	1968-11-12	Marketing

A3

Index	Member
1	[[[Luis,M,1968-11-27, ...],[Olivia,F,1968-11-05, ...],[Nichola...
2	[[[David,M,1969-01-31, ...],[Tyler,M,1969-01-24, ...],[Jacob,...
3	[[[Courtney,F,1970-01-27, ...]],[[Lauren,F,1970-02-06, ...]],[[...
4	[[[Emily,F,1971-01-08, ...],[Ryan,M,1971-01-13, ...],[Madiso...
5	[[[Jaco...

Index	Member
1	[[Luis,M,1968-11-27, ...],[Olivia,F,1968-11-05, ...],[Nichola...
2	[[Sarah,F,1968-12-29, ...],[Tyler,M,1968-12-23, ...]]

Index	NAME	GENDER	BIRTHDAY	DEPT
1	Luis	M	1968-11-27	Technology
2	Olivia	F	1968-11-05	Production
3	Nicholas	M	1968-11-24	Marketing
4	Alexis	F	1968-11-12	Marketing

Grouped subsets need to be retained when routine aggregation is difficult to compute targets

Example: Check the information of the two oldest people in each department.

	A	B
1	=demo.query("select * from EMPLOYEE")	
2	=A1.group(DEPT).(~.top(2;-age(BIRTHDAY))).conj()	/Find the oldest two people in the subset

A1

Index	EID	NAME	SURNAME	GENDER	STATE	BIRTHDAY	HIREDATE	DEPT	SALARY
1	1	Rebecca	Moore	F	California	1974-11-20	2005-03-11	R&D	7000
2	2	Ashley	Wilson	F	New York	1980-07-19	2008-03-16	Finance	11000
3	3	Rachel	Johnson	F	New Mexico	1970-12-17	2010-12-01	Sales	9000
4	4	Emily	Smith	F	Texas	1985-03-07	2006-08-15	HR	7000
5	5	Ashley	Smith	F	Texas	1975-05-13	2004-07-30	R&D	16000
6	6	Matthew	Johnson	M	California	1984-07-07	2005-07-07	Sales	11000
7	7	Alexis	Smith	F	Illinois	1972-08-16	2002-08-16	Sales	9000
8	8	Megan	Wilson	F	California	1979-04-19	1984-04-19	Marketing	11000

A2

Index	EID	NAME	SURNAME	GENDER	STATE	BIRTHDAY	HIREDATE	DEPT	SALARY
1	18	Jonathan	Moore	M	Florida	1971-03-07	2000-03-07	Administrat...	7000
2	20	Alexis	Allen	F	Florida	1977-08-07	2007-08-07	Administrat...	16000
3	242	Ashley	Jones	F	North Caro...	1972-08-07	2001-09-01	Finance	6500
4	240	Jacob	Thomas	M	Texas	1972-01-10	2009-06-01	Finance	7000
5	174	Michael	Miller	M	Wisconsin	1971-07-25	2001-01-01	HR	10000
6	162	Gabriel	Wilson	M	California	1971-12-12	2001-10-01	HR	12000
7	444	Alexis	Johnson	F	New Hamp...	1968-11-12	2010-12-01	Marketing	6500
8	440	Nicholas	Smith	M	Illinois	1968-11-24	2008-07-01	Marketing	8000

Example: Calculate the login times of each account within three days before the last login time

	A
1	=file("E:/txt/login.txt").import@t()
2	=A1.group(userid;~.max(login_time):last,~.count(interval(login_time,last)<=3):login_num)
	/Grouping by userid, and then calculating the last login time and the number of logins within three days before that time

A1

Index	userid	login_time
1	1	2019-05-01 03:06:59
2	1	2019-05-01 16:05:49
3	1	2019-05-02 03:29:35
4	1	2019-05-02 14:56:25
5	1	2019-05-02 16:43:11
6	1	2019-05-02 18:29:48
7	1	2019-05-02 23:39:06
8	1	2019-05-03 02:10:27
9	1	2019-05-03 09:06:08
10	1	2019-05-03 14:41:56

A2

Index	userid	last	login_num
1	1	2019-05-05 ...	14
2	2	2019-05-05 ...	10
3	3	2019-05-05 ...	7
4	4	2019-05-04 ...	6
5	5	2019-05-05 ...	7

Grouped subsets can be used for sorting. When there are many duplicate members, group@s(x) sorting is faster than sort.

Example: Sort employees by department

	A
1	=demo.query("select * from EMPLOYEE")
2	=A1.group(DEPT).conj()
3	=A1.group@s(DEPT)

A1

Index	EID	NAME	GENDER	DEPT
1	1	Rebecca	F	R&D
2	2	Ashley	F	Finance
3	3	Rachel	F	Sales
4	4	Emily	F	HR
5	5	Ashley	F	R&D
6	6	Matthew	M	Sales
7	7	Alexis	F	Sales
8	8	Megan	F	Marketing
9	9	Victoria	F	HR
10	10	Ryan	M	R&D

A2,A3

Index	EID	NAME	GENDER	DEPT
1	18	Jonathan	M	Administrat...
2	20	Alexis	F	Administrat...
3	26	Timothy	M	Administrat...
4	42	Michael	M	Administrat...
5	2	Ashley	F	Finance
6	13	Daniel	M	Finance
7	23	Joseph	M	Finance
8	24	Chloe	F	Finance
9	32	Andrew	M	Finance
10	217	Emily	F	Finance

02

Order-related grouping

Let's start with a simple example of dividing company members into three groups on average.

	A	B
1	=demo.query("select EID,NAME,GENDER,STATE from EMPLOYEE")	
2	=A1.group((#-1)*3\A1.len())	/The first 1/3, the middle 1/3 and the last 1/3
3	=A1.group((#-1)%3)	/Take one out of every three members

A1~A3 results:

Index	EID	NAME	GENDER	STATE
1	1	Rebecca	F	California
2	2	Ashley	F	New York
3	3	Rachel	F	New Mexico
4	4	Emily	F	Texas
5	5	Ashley	F	Texas
6	6	Matthew	M	California
7	7	Alexis	F	Illinois
8	8	Megan	F	California
9	9	Victoria	F	Texas
10	10	Ryan	M	Pennsylva...

Index	Member
1	[[1,Rebecca,F, ...],[2,Ashley,F, ...],[3,Rachel,F, ...], ...]
2	[[168,Nicholas,M, ...],[169,Hannah,F, ...],[170,Nicholas,M,...
3	[[335,Madison,F, ...],[336,Ryan,M, ...],[337,Lauren,F, ...], ...]

Index	EID	NAME	GENDER	STATE
1	168	Nicholas	M	New York
2	169	Hannah	F	Connecticut
3	170	Nicholas	M	Tennessee
4	171	Megan	F	California
5	172	Brandon	M	Pennsylvania

Index	Member
1	[[1,Rebecca,F, ...],[4,Emily,F, ...],[7,Alexis,F, ...], ...]
2	[[2,Ashley,F, ...],[5,Ashley,F, ...],[8,Megan,F, ...], ...]
3	[[3,Rachel,F, ...],[6,Matthew,M, ...],[9,Victoria,F, ...], ...]

Index	EID	NAME	GENDER	STATE
1	1	Rebecca	F	California
2	4	Emily	F	Texas
3	7	Alexis	F	Illinois
4	10	Ryan	M	Pennsylvania
5	13	Daniel	M	Florida

The format of the existing log is as follows:

The first row is IP, TIME, GET, URL, BROWER;

The second row is MODULE;

The third row is USERID, UNAME, LOCATION: Please organize this log into structured data.

```
10.10.10.143        2013-04-01 21:14:44        GET        /p/pt301/index.jsp  Mozilla/6.0
#module:production#
47356    Jessica    Chicago
10.10.2.76          2013-04-01 21:18:50        GET        /h/homepage.jsp    Chrome/35
#module:homepage#
419      Jacob      Houston
10.10.54.218        2013-04-01 21:25:19        GET        /p/pt27/index.jsp   Mozilla/6.0
#module:production#
3464     Madison    Detroit
10.10.10.145        2013-04-01 21:30:02        GET        /u/userlist/showlist.jsp        Mozilla/6.0
#module:usercenter#
432442   Phoenix    San Jose
10.2.1.242          2013-04-01 21:30:15        GET        /p/pt271/index.jsp  Mozilla/6.0
#module:production#
3435567  Megan      San Jose
```

Processing non-single-row logs

Order-related grouping

	A	B
1	=file("E:/txt/access_log.txt").import@s()	
2	=A1.group((#-1)\3)	/Grouping every three rows
3	=A2.(~.(_1).concat("\t").array("\t"))	/Merge the members of each group and then split them into field values with "\t"
4	=A3.new(~(7):USERID,~(8):UNAME,~(1):IP,~(2):TIME,~(4):URL,~(5):BROWER,~(9):LOCATION,left(~(6).array("\:")(2),-1):module)	/Structuring the results

A1

Index	_1
1	10.10.10.1432013-04-01 21:14:44GET/p/pt301/index.js...
2	#module:production#
3	47356JessicaChicago
4	10.10.2.762013-04-01 21:18:50GET/h/homepage.jspC...
5	#module:homepage#
6	419JacobHouston
7	10.10.54.2182013-04-01 21:25:19GET/p/pt27/index.jsp...
8	#module:production#
9	3464MadisonDetroit
10	10.10.10.1452013-04-01 21:30:02GET/u/userlist/showli...

A2

Index	Member
1	[[10.10.10.14320 04-01 21:14:44GET/p/pt301/index.jspMozilla/6....
2	[[10.10.2.762013-04- 21:18:50GET/h/homepage.jspChrome/35],[...
3	[[10.10.54.2182013-04-0 1:25:19GET/p/pt27/index.jspMozilla/6.0]...
4	[[10.10.10.1452013-04-01 21 :02GET/u/userlist/showlist.jspMozil...
5	[[10.2.1.2422013-04-01 21:30:15 T/p/pt271/index.jspMozilla/6.0],[...

Index	_1
1	10.10.10.1432013-04-01 21:14:44GET/p/pt301/index.jspMozilla/6.0
2	#module:production#
3	47356JessicaChicago

A3

Index	Member
1	[10.10.1 143,2013-04-01 21:14:44,GET, ...]
2	[10.10.2.7 013-04-01 21:18:50,GET, ...]
3	[10.10.54.218, 13-04-01 21:25:19,GET, ...]
4	[10.10.10.145,201 4-01 21:30:02,GET, ...]
5	[10.2.1.242,2013-04-0 1:30:15,GET, ...]

Index	Member
1	10.10.10.143
2	2013-04-01 21:14:44
3	GET
4	/p/pt301/index.jsp
5	Mozilla/6.0
6	#module:production#
7	47356
8	Jessica
9	Chicago

A4

Index	USERID	UNAME	IP	TIME	URL	BROWER	LOCATION	module
1	47356	Jessica	10.10.10.1...	2013-04-0...	/p/pt301/in...	Mozilla/6.0	Chicago	production
2	419	Jacob	10.10.2.76	2013-04-0...	/h/homepa...	Chrome/35	Houston	homepage
3	3464	Madison	10.10.54.2...	2013-04-0...	/p/pt27/ind...	Mozilla/6.0	Detroit	production
4	432442	Phoenix	10.10.10.1...	2013-04-0...	/u/userlist/...	Mozilla/6.0	San Jose	usercenter
5	3435567	Megan	10.2.1.242	2013-04-0...	/p/pt271/in...	Mozilla/6.0	San Jose	production

Grouping by Boolean Value

Example: The students are divided into two classes according to their ranking, so that the average ranking of the two classes is the same.

	A	B
1	=file("E:/txt/students_score.txt").import@t()	
2	=A1.sort@z(sum(~.array()))	/Sort by scores
3	=A2.group((#-1)%4<2)	/Rank [1,4,5,8...] [2,3,6,7...], divide into two groups

A1~A3 results:

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Zachary	75	81	85
6	Sophia	74	86	93
7	Hannah	90	76	95
8	Christopher	71	81	86
9	Sean	98	86	81
10	Tyler	87	78	93

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Zachary	75	81	85
6	Sophia	74	86	93
7	Hannah	90	76	95
8	Christopher	71	81	86
9	Sean	98	86	81
10	Tyler	87	78	93

Index	Member
1	[[Hannah,90,76, ...],[Natalie,84,90, ...],[Jessica,87,88,...
2	[[Emma,88,84, ...],[Sean,98,86, ...],[Tyler,87,78, ...], ...]

Index	Name	Math	Chinese	English
1	Hannah	90	76	95
2	Natalie	84	90	84
3	Jessica	87	88	78
4	Sophia	74	86	93

When the data itself is ordered, @o option is sorted by key values and is faster than regular grouping

Example: Count of the total length of time users listen to music (data in UID order)

	A
1	=file("E:/txt/music_watch.txt").import@t()
2	=now()
3	=A1.groups(uid;sum(watch_time):sum_time)
4	=interval@ms(A2,now())
5	=now()
6	=A1.groups@o(uid;sum(watch_time):sum_time)
7	=interval@ms(A5,now())

A1 (321039 rows)

Index	uid	musicid	watch_time	time
1	100000	608909109	270	13:39:05
2	100000	329900206	57	20:55:19
3	100000	177200319	75	22:45:21
4	100000	9773909220	291	16:59:36
5	100000	2851509115	75	18:49:21
6	100000	969900330	118	13:55:07
7	100000	6692209167	187	16:39:38
8	100000	571700343	287	11:54:47
9	100000	603009659	127	09:05:42
10	100000	8632109316	101	14:45:19

A3, A6 results:

Index	uid	sum_time
1	100000	1588
2	100001	1255
3	100002	1718
4	100003	1502
5	100004	2110
6	100005	1014
7	100006	1156
8	100007	2305
9	100008	1739
10	100009	3000

A4

Value
32

A7

Value
24

Grouping by Value: Find out the team record with the most consecutive championships in NBA history

	A	B
1	=file("E:/txt/NBA.txt").import@t().sort(year)	
2	=A1.group@o(champion).maxp(~.len())	/Regrouping when champion field changes

A1, A2 results:

Index	Year	Date	Champion	Total_score	Runner_up
1	1947	4.16-4.22	Philadelphia W...	4-1	Chicago Stags
2	1948	4.10-4.21	Baltimore Bull...	4-2	Philadelphia W...
3	1949	4.4-4.13	Minneapolis L...	4-2	Washington C...
4	1950	4.8-4.23	Minneapolis L...	4-2	Serakus Natio...
5	1951	4.7-4.21	Royal Rochester	4-3	New York Knicks
6	1952	4.12-4.25	Minneapolis L...	4-3	New York Knicks
7	1953	4.4-4.10	Minneapolis L...	4-1	New York Knicks
8	1954	3.31-4.12	Minneapolis L...	4-3	Serakus Natio...
9	1955	3.31-4.10	Serakus Natio...	4-3	Ford Wayne Pi...
10	1956	3.31-4.7	Philadelphia W...	4-1	Ford Wayne Pi...

Index	Year	Date	Champion	Total_score	Runner_up
1	1959	4.4-4.9	Boston Celtics	4-0	Minneapolis L...
2	1960	3.27-4.9	Boston Celtics	4-3	St. Louis Hawks
3	1961	4.2-4.11	Boston Celtics	4-1	St. Louis Hawks
4	1962	4.7-4.18	Boston Celtics	4-3	Los Angeles L...
5	1963	4.14-4.24	Boston Celtics	4-2	Los Angeles L...
6	1964	4.18-4.26	Boston Celtics	4-1	San Francisco ...
7	1965	4.18-4.25	Boston Celtics	4-1	Los Angeles L...
8	1966	4.17-4.28	Boston Celtics	4-3	Los Angeles L...

Processing indefinite-row text

The following is the user information of a website, which includes: userid, gender, age, salary, province, musicid, watch_time, time.

The information of each user is incomplete.

Please organize the following data into structured data.

```
userid:00ea9a2fe9c6810aab440c4d8c050000    gender:F  age:26
userid:00ea9a2fe9c6810aab440c4d8c050000    salary:20000-100000
userid:00ea9a2fe9c6810aab440c4d8c050000    musicid:4090309101
userid:00ea9a2fe9c6810aab440c4d8c050000    watch_time:15      time:2019-05-04 19:05:45
userid:01d86fc1401b283d5828c293be290e08    gender:Mage:18
userid:01d86fc1401b283d5828c293be290e08    salary:0-2000
userid:01d86fc1401b283d5828c293be290e08    province:江苏
userid:01d86fc1401b283d5828c293be290e08    musicid:6192809101
userid:01d86fc1401b283d5828c293be290e08    watch_time:385     time:2019-05-04 12:36:11
userid:002f4b9c49be9a0b2c13e1c3c4f6a21c    gender:F  age:28
userid:002f4b9c49be9a0b2c13e1c3c4f6a21c    province:黑龙江
userid:002f4b9c49be9a0b2c13e1c3c4f6a21c    musicid:6192809101
userid:002f4b9c49be9a0b2c13e1c3c4f6a21c    watch_time:73      time:2019-05-04 22:54:29
```

Processing indefinite-row text

	A	B
1	=file("E:/txt/Indefinite _info.txt").import@s()	
2	[userid,gender,age,salary,province,musicid,watch_time,time]	
3	=A1.group@o(_1.array("\t")(1))	/Grouping according to changes in values
4	=A3.(~.(_1.array("\t")).conj().id().align(A2,~.array("\:")(1)).(~.array("\:")(2))).conj()	/Intra-group processing
5	=create(${A2.concat@c()}).record(A4)	/Create table and fill in records

A1

Index	_1
1	userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:26
2	userid:00ea9a2fe9c6810aab440c4d8c050000salary:20000-100000
3	userid:00ea9a2fe9c6810aab440c4d8c050000musicid:4090309101
4	userid:00ea9a2fe9c6810aab440c4d8c050000watch_time:15time:2019-05-04 ...
5	userid:01d86fc1401b283d5828c293be290e08gender:Mage:18
6	userid:01d86fc1401b283d5828c293be290e08salary:0-2000
7	userid:01d86fc1401b283d5828c293be290e08province:jiangsu
8	userid:01d86fc1401b283d5828c293be290e08musicid:6192809101
9	userid:01d86fc1401b283d5828c293be290e08watch_time:385time:2019-05-0...
10	userid:002f4b9c49be9a0b2c13e1c3c4f6a21cgender:Fage:28
11	userid:002f4b9c49be9a0b2c13e1c3c4f6a21cprovince:heilongjiang
12	userid:002f4b9c49be9a0b2c13e1c3c4f6a21cmusicid:6192809101
13	userid:002f4b9c49be9a0b2c13e1c3c4f6a21cwatch_time:73time:2019-05-04 2...

A3

Index	Member
1	[[userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:26],[userid:00...
2	[[userid:01d86fc1401b283d5828c293be290e08gender:Mage:18],[userid:0
3	[[u

Index	_1
1	userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:26
2	userid:00ea9a2fe9c6810aab440c4d8c050000salary:20000-100000
3	userid:00ea9a2fe9c6810aab440c4d8c050000musicid:4090309101
4	userid:00ea9a2fe9c6810aab440c4d8c050000watch_time:15time:2019-05...

A4

Index	Member
1	00ea9a2fe9c6810aab440c4d8c050000
2	F
3	26
4	20000-100000
5	(null)
6	4090309101
7	15
8	2019-05-04 19
9	01d86fc1401b283d5828c293be290e08
10	M

A5

Index	userid	gender	age	salary	province	musicid	watch_time	time
1	00ea9a2fe...	F	26	20000-100...	(null)	4090309101	15	2019-05-04 19
2	01d86fc14...	M	18	0-2000	jiangsu	6192809101	385	2019-05-04 12
3	002f4b9c4...	F	28	(null)	heilongjiang	6192809101	73	2019-05-04 22

Processing indefinite-row text(2)

Userid is recorded in the first row of the user's information. The following rows are the user's information. When the next userid appears, it will be the next user's information:

```
userid:00ea9a2fe9c6810aab440c4d8c050000      gender:F  age:26
salary:20000-100000
musicid:4090309101
watch_time:15        time:2019-05-04 19:05:45
userid:01d86fc1401b283d5828c293be290e08      gender:Mage:18
salary:0-2000
province:江苏
musicid:6192809101
watch_time:385       time:2019-05-04 12:36:11
userid:002f4b9c49be9a0b2c13e1c3c4f6a21c      gender:F  age:28
province:黑龙江
musicid:6192809101
watch_time:73        time:2019-05-04 22:54:29
```

Processing indefinite-row text(2)

	A	B
1	=file("E:/txt/Indefinite _info2.txt").import@s()	
2	[userid,gender,age,salary,province,musicid,watch_time,time]	
3	=A1.group@i(_1.array("\t")(1).array("\:")(1)=="userid")	/Grouping according to changes in values
4	=A3.(~.(_1.array("\t")).conj().align(A2,~.array("\:")(1)).(~.array("\:")(2))).conj()	/Intra-group processing
5	=create(${A2.concat@c()}).record(A4)	/Create table and fill in records

A1

Index	_1
1	userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:26
2	salary:20000-100000
3	musicid:4090309101
4	watch_time:15time:2019-05-04 19:05:45
5	userid:01d86fc1401b283d5828c293be290e08gender:Mage:18
6	salary:0-2000
7	province:江苏
8	musicid:6192809101
9	watch_time:385time:2019-05-04 12:36:11
10	userid:002f4b9c49be9a0b2c13e1c3c4f6a21cgender:Fage:28
11	province:黑龙江
12	musicid:6192809101
13	watch_time:73time:2019-05-04 22:54:29

A3

Index	Member
1	[[userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:2...
2	[[userid:01d86fc1401b283d5828c293be290e08gender:Mage:1...
3	[[userid:002f4b9c49be9a0b2c13e1c3c4f6a21cgender:Fage:28]...

Index	_1
1	userid:00ea9a2fe9c6810aab440c4d8c050000gender:Fage:26
2	salary:20000-100000
3	musicid:4090309101
4	watch_time:15time:2019-05-04 19:05:45

A4

Index	Member
1	00ea9a2fe9c6810aab440c4d8c050000
2	F
3	26
4	20000-100000
5	(null)
6	4090309101
7	15
8	2019-05-04 19
9	01d86fc1401b283d5828c293be290e08
10	M

A5

Index	userid	gender	age	salary	province	musicid	watch_time	time
1	00ea9a2fe...	F	26	20000-100...	(null)	4090309101	15	2019-05-04 19
2	01d86fc14...	M	18	0-2000	江苏	6192809101	385	2019-05-04 12
3	002f4b9c4...	F	28	(null)	黑龙江	6192809101	73	2019-05-04 22

Example: Calculate the maximum number of consecutive days for stock1001 price to rise

	A	B
1	=file("E:/txt/stock1001_price.txt").import@t() .sort(DT)	
2	=A1.group@i(CL<CL[-1])	/If CL<CL[-1] is True, create a new group
3	=A2.max(~.len())	/Find out the maximum value of group members

A1~A3 results:

Index	stockid	DT	CL
1	1001	2009-01-01	4.0
2	1001	2009-01-02	3.64
3	1001	2009-01-05	3.95
4	1001	2009-01-06	3.68
5	1001	2009-01-07	3.53
6	1001	2009-01-08	3.59
7	1001	2009-01-09	3.9
8	1001	2009-01-12	3.56
9	1001	2009-01-13	3.22
10	1001	2009-01-14	3.17

Index	Member
1	[[1001,2009-01-01,4.0]]
2	[[1001,2009-01-02,3.64],[1001,2009-01-05,3.95]]
3	[[1001,2009-01-06,3.68]]
4	[[1001,2009-01-07,3.53],[1001,2009-01-08,3.59],[100...
5	[[1001,2009-01-12,3.56]]
6	[[1001,2009-01-13,3.22]]
7	[[1001,2009-01-14,3.17]]
8	[[1001,2009-01-15,3.15],[1001,2009-01-16,3.46],[100...
9	[[1001,2009-01-21,3.87],[1001,2009-01-22,4.13],[100...
10	[[1001,20
11	[[1001,20

Index	stockid	DT	CL
1	1001	2009-01-02	3.64
2	1001	2009-01-05	3.95

Value
4

When the group key value is ordinal, grouping with the @n option (including group, groups) is faster than normal grouping because hash values do not need to be calculated.

Example: Calculate total monthly sales

	A
1	=file("E:/txt/orders_mm.txt").import@t()
2	=now()
3	=A1.groups(month;sum(amount))
4	=interval@ms(A2,now())
5	=now()
6	=A1.groups@n(month;sum(amount))
7	=interval@ms(A5,now())

A1 （10 million rows）

Index	sellerid	amount	month
9999995	10122	1638.8283...	12
9999996	10170	1797.2074...	12
9999997	10876	509.11425...	12
9999998	10767	1562.8705...	12
9999999	10327	966.58393...	12
10000000	10711	1497.0917...	12

A3, A6 results:

Index	month	sum(amount)
1	1	1.0414380084365...
2	2	1.0409466247907...
3	3	1.0399434822629...
4	4	1.0405603690967...
5	5	1.0418574292546...

A4

Value
1081

A7

Value
504

When the key value to be sorted is ordinal, it can also be sorted by grouping with the @n option, which is faster than sort.

Example: Ranking employees according to their age

	A	B
1	=demo.query("select EID,NAME,GENDER,BIRTHDAY from EMPLOYEE")	
2	=A1.group@n(age(BIRTHDAY)\10).conj()	/Grouping by year, then merge

A1, A2 results:

Index	EID	NAME	GENDER	BIRTHDAY
1	1	Rebecca	F	1974-11-20
2	2	Ashley	F	1980-07-19
3	3	Rachel	F	1970-12-17
4	4	Emily	F	1985-03-07
5	5	Ashley	F	1975-05-13
6	6	Matthew	M	1984-07-07
7	7	Alexis	F	1972-08-16
8	8	Megan	F	1979-04-19
9	9	Victoria	F	1983-12-07
10	10	Ryan	M	1976-03-12

Index	EID	NAME	GENDER	BIRTHDAY
1	2	Ashley	F	1980-07-19
2	4	Emily	F	1985-03-07
3	6	Matthew	M	1984-07-07
4	9	Victoria	F	1983-12-07
5	12	Jessica	F	1980-09-11
6	13	Daniel	M	1982-05-14
7	15	Alexis	F	1983-07-10
8	17	Hannah	F	1980-07-19
9	22	Jacob	M	1985-05-07
10	23	Joseph	M	1983-08-27

03

Order-related aggregation

Order-related situations of intragroup aggregation

Example: Calculate each person's starting and ending duty time

	A	B
1	=file("E:/txt/duty.txt").import@t().sort(date)	
2	=A1.group@o(name)	/Grouping in data order (Regroup when data changes)
3	=A2.new(name,~.m(1).date:begin,~.m(-1).date:end)	/The first as the beginning and the last as the end.

A1~A3 results:

Index	date	name
1	2018-03-01	Emily
2	2018-03-02	Emily
3	2018-03-03	Emily
4	2018-03-04	Johnson
5	2018-03-05	Ashley
6	2018-03-06	Emily
7	2018-03-07	Emily
8	2018-03-08	Ashley
9	2018-03-09	Emily
10	2018-03-10	Ashley

Index	Member
1	[[2018-03-01,Emily],[2018-03-0...
2	[[2018-03-04,Johnson]]
3	[[2018-03-05,Ashley]]
4	[[2018-03-06,Emily],[2018-0...-0...
5	[[2018-03-08,Ashley]]

Index	date	name
1	2018-03-01	Emily
2	2018-03-02	Emily
3	2018-03-03	Emily

Index	name	begin	end
1	Emily	2018-03-01	2018-03-03
2	Johnson	2018-03-04	2018-03-04
3	Ashley	2018-03-05	2018-03-05
4	Emily	2018-03-06	2018-03-07
5	Ashley	2018-03-08	2018-03-08
6	Emily	2018-03-09	2018-03-09
7	Ashley	2018-03-10	2018-03-10
8	Johnson	2018-03-11	2018-03-15
9	Ashley	2018-03-16	2018-03-16
10	Johnson	2018-03-17	2018-03-17

Example: Calculate the login times of each account within three days before the last login time. Previously, we have done it with the conventional subset operation. Here, we use the ordered feature of subsets to complete it.

	A
1	=file("E:/txt/login.txt").import@t().sort@z(login_time)
2	=A1.group(userid;~(1).login_time:last,~.pselect@n(interval(login_time,last)>3)-1:num)
	/Sort according to the reverse order of login time, the login time of the first member after grouping is the last login time, and then calculate the number of logins in three days.

A1

Index	userid	login_time
1	2	2019-05-05 22:16:39
2	1	2019-05-05 17:24:32
3	3	2019-05-05 17:04:51
4	3	2019-05-05 14:10:17
5	1	2019-05-05 05:38:55
6	5	2019-05-05 05:28:58
7	2	2019-05-05 02:58:38
8	5	2019-05-04 22:09:13
9	2	2019-05-04 22:07:15
10	2	2019-05-04 20:58:35

A2

Index	userid	last	num
1	1	2019-05-05 1...	14
2	2	2019-05-05 2...	10
3	3	2019-05-05 1...	7
4	4	2019-05-04 0...	6
5	5	2019-05-05 0...	7

Example: Calculate the daily increase of the highest price of each stock
(if the first day is the highest price, record the price)

	A
1	=file("E:/txt/stock_price.txt").import@t().sort(DT)
2	=A1.group(stockid;(p=~.pmax(CL),if(p==1,~.CL,~.calc(p,CL/CL[-1]-1))):rises)
	/Groups according to stockid, find the position of the highest price and assign it to p, then calculate the increase.

A1

Index	stockid	DT	CL
1	1001	2009-01-01	4.0
2	1026	2009-01-01	2.13
3	1028	2009-01-01	20.09
4	1070	2009-01-01	14.95
5	1107	2009-01-01	3.74
6	1134	2009-01-01	10.68
7	1137	2009-01-01	42.31
8	1147	2009-01-01	19.61
9	1206	2009-01-01	14.01
10	1213	2009-01-01	40.94

A2

Index	stockid	rises
1	1001	0.09691629955947145
2	1026	0.04958677685950419
3	1028	0.0155920775389801...
4	1070	0.07007203667321549
5	1107	0.09358288770053469
6	1134	0.0280199252801993...
7	1137	0.05385810460901075
8	1147	0.0014598540145984...
9	1206	14.01
10	1213	40.94

Example: Count how many months it took for salesman to break through 500,000 sales

	A
1	=file("E:/txt/orders_i.csv").import@t()
2	=A1.group(sellerid;(~.iterate((x=#,~~=~~+amount),0,~~>500000),x):breach50)
	/Group according to sellerid, the amount is accumulated until it is more than 500000, when x is the number of months needed.

A1

Index	sellerid	month	amount
1	1	1	75916.4
2	1	2	62083.7
3	1	3	119098.6
4	1	4	145296.6
5	1	5	87776.0
6	1	6	76660.3
7	1	7	74950.6
8	1	8	55814.5
9	1	9	138220.5
10	1	10	49445.5

A2

Index	sellerid	breach50
1	1	6
2	2	8
3	3	7
4	4	6
5	5	8
6	6	10
7	7	9
8	8	8
9	9	7
10	10	7

Example: Calculate the highest score of each class

	A
1	=file("E:/txt/students_subject.txt").import@t()
2	=A1.group(CLASS;~.max(iterate(~~+SCORE,0;STUDENTID)):max_score)
	/Group according to CLASS, the SCORE is accumulated. When STUDENTID changes, the calculation is restarted, and then the maximum value in the group is taken.

A1

Index	CLASS	STUDENTID	SUBJECT	SCORE
1	Class one	1	English	84
2	Class one	1	Math	77
3	Class one	1	PE	69
4	Class one	2	English	81
5	Class one	2	Math	80
6	Class one	2	PE	97
7	Class one	3	English	75
8	Class one	3	Math	86
9	Class one	3	PE	67
10	Class one	4	English	96

A2

Index	CLASS	max_score
1	Class one	258
2	Class two	252

Example: The monthly interest rate of a credit company changes with time.

Calculate the amount of repayments due to each user.

	A
1	=file("E:/txt/rate.txt").import@t()
2	=A1.groups(userid;iterate(~~*(1+rate_m),principal):repayment)
	/Group according to userid, repayment = principal * (1 + January interest rate)* (1 + February interest rate)*(...).

A1

Index	userid	principal	month	rate_m
1	1	10000	1	0.007313
2	1	10000	2	0.007275
3	1	10000	3	0.007259
4	1	10000	4	0.007248
5	1	10000	5	0.00724
6	2	30000	1	0.007313
7	2	30000	2	0.007275
8	2	30000	3	0.007259
9	2	30000	4	0.007248
10	2	30000	5	0.00724

A2

Index	userid	repayment
1	1	10368.669428018195
2	2	31106.008284054584

Example: Ranking students in different classes according to their scores

	A	
1	=file("E:/txt/students_c.txt").import@t().sort(CLASS,-SCORE)	/Sort in reverse order according to CLASS,SCORE
2	=A1.derive(rank)	/Add a new column rank
3	>A2.group(CLASS).(~.run(iterate(if(SCORE==SCORE[-1],~~,#)):rank))	/Group by class， Use run function and iterate function to assign values to rank column

A1

Index	CLASS	NAME	SCORE
1	1	Smith Willi...	629
2	1	Garcia Bryan	628
3	1	Jones Justin	628
4	1	Lee Rachel	628
5	1	Moore Mich...	628
6	1	Smith Reb...	628
7	1	Lewis Gab...	627
8	1	Martin Jos...	627
9	1	Moore Jon...	627
10	1	Williams N...	627

A2 when A2 executed

Index	CLASS	NAME	SCORE	rank
1	1	Smith Willi...	629	(null)
2	1	Garcia Bryan	628	(null)
3	1	Jones Justin	628	(null)
4	1	Lee Rachel	628	(null)
5	1	Moore Mich...	628	(null)
6	1	Smith Reb...	628	(null)
7	1	Lewis Gabr...	627	(null)
8	1	Martin Jose...	627	(null)
9	1	Moore Jona...	627	(null)
10	1	Williams N...	627	(null)

A2 after A3 executed

Index	CLASS	NAME	SCORE	rank
1	1	Smith Willi...	629	1
2	1	Garcia Bryan	628	2
3	1	Jones Justin	628	2
4	1	Lee Rachel	628	2
5	1	Moore Mich...	628	2
6	1	Smith Reb...	628	2
7	1	Lewis Gabr...	627	7
8	1	Martin Jose...	627	7
9	1	Moore Jona...	627	7
10	1	Williams N...	627	7

04

Enumeration grouping

The preceding examples share two common features:

1. Grouping results have no empty subset
2. Any member of the original set belongs to and belongs only to a subset.

We call this kind of partition a complete partition.

SPL provides functions such as align, enum, etc. which are not completely partitioned.

Let's look at the SPL feature grouping.

Count the number of employees under 40 years old in all departments of the company

	A	B
1	=demo.query("select * from EMPLOYEE")	
2	=A1.id(DEPT)	/List all departments
3	=A1.select(age(BIRTHDAY)<40).align@a(A2,DEPT)	/Employees younger than 40 years old were screened and grouped according to A2
4	=A3.new(A2(#):DEPT,count(~):num_lt40)	/ Count the number of people under 40 in various departments

A1~A4 results:

Index	EID	NAME	BIRTHDAY	DEPT
1	1	Rebecca	1974-11-20	R&D
2	2	Ashley	1980-07-19	Finance
3	3	Rachel	1970-12-17	Sales
4	4	Emily	1985-03-07	HR
5	5	Ashley	1975-05-13	R&D
6	6	Matthew	1984-07-07	Sales
7	7	Alexis	1972-08-16	Sales
8	8	Megan	1979-04-19	Marketing
9	9	Victoria	1983-12-07	HR
10	10	Ryan	1976-03-12	R&D

Index	Member
1	Administration
2	Finance
3	HR
4	Marketing
5	Production
6	R&D
7	Sales
8	Technology

Index	Member
1	[]
2	[[2,Ashley,1980-07-19, ...],[13,Daniel,198...
3	[[4,Emily,1985-03-07, ...],[9,Victoria,1983-...
4	[[17,Hanna...
5	[[28,Zachar...
6	[[22,Jacob,...
7	[[6,Matthew...
8	

Index	Member

Index	EID	NAME	BIRTHDAY	DEPT
1	2	Ashley	1980-07-19	Finance
2	13	Daniel	1982-05-14	Finance
3	23	Joseph	1983-08-27	Finance
4	218	Kayla	1984-12-18	Finance
5	219	Emily	1986-10-25	Finance

Index	DEPT	num_lt40
1	Administration	0
2	Finance	15
3	HR	8
4	Marketing	33
5	Production	43
6	R&D	9
7	Sales	82
8	Technology	20

Check all the scores of the top three English scorers

	A	B
1	=file("E:/txt/student_scores.txt").import@t()	
2	=A1.select(SUBJECT=="English").top(3;-SCORE)	/Find the records of the top three English scorers
3	=A2.(STUDENTID)	/List the StudentID
4	=A1.align@a(A3,STUDENTID).conj()	/Grouping A1 in sequence of A3, and finally merging sets

A1~A4 results:

Index	STUDENTID	SUBJECT	SCORE
1	1	English	84
2	1	Math	77
3	1	PE	69
4	2	English	81
5	2	Math	80
6	2	PE	97
7	3	English	75
8	3	Math	86
9	3	PE	67
10	4	English	96
11	4	Math	63
12	4	PE	81

Index	STUDENTID	SUBJECT	SCORE
1	4	English	96
2	1	English	84
3	2	English	81

Index	Member
1	4
2	1
3	2

Index	STUDENTID	SUBJECT	SCORE
1	4	English	96
2	4	Math	63
3	4	PE	81
4	1	English	84
5	1	Math	77
6	1	PE	69
7	2	English	81
8	2	Math	80
9	2	PE	97

Count the number of students whose mathematic scores are less than or equal to 80, greater than 80, less than or equal to 90, and greater than 90, respectively

	A	B
1	=file("E:/txt/student_scores.txt").import@t()	
2	[?<=80,?>80&&?<=90,?>90]	
3	=A1.enum(A2,Math)	/Enumeration grouping

A1, A3 results:

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Zachary	75	81	85
6	Sophia	74	86	93
7	Hannah	90	76	95
8	Christopher	71	81	86
9	Sean	98	86	81
10	Tyler	87	78	93

Index	Member
1	[[Zachary,75,81, ...],[Sophia,74,8],[Christopher,71,...
2	[[Natalie,84,90, ...],[Jessica,87,88, ...],[Br na,89,90, ...
3	[[Sean,98,86, ...]]

Index	Name	Math	Chinese	English
1	Zachary	75	81	85
2	Sophia	74	86	93
3	Christopher	71	81	86

Count the number of students whose mathematic scores are less than or equal to 80, greater than 80, and greater than 90, respectively

	A	B
1	=file("E:/txt/student_scores.txt").import@t()	
2	[?<=80,?>80,?>90]	
3	=A1.enum(A2,Math)	/Enumeration grouping
4	=A1.enum@r(A2,Math)	/Members appear in different groups

A1, A3, A4 results

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Zachary	75	81	85
6	Sophia	74	86	93
7	Hannah	90	76	95
8	Christopher	71	81	86
9	Sean	98	86	81
10	Tyler	87	78	93

Index	Member

Group that are greater than 90

Index	Member
1	[Zachary,75,81, ...],[Sophia,74,86, ...],[Christopher,71,...
2	[[Natalie,84,90, ...],[Jessica,87,88, ...],[Brianna,89,90, ...
3	[]

Group that are greater than 80

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Hannah	90	76	95
6	Sean	98	86	81
7	Tyler	87	78	93

Index	Name	Math	Chinese	English
1	Sean	98	86	81

Group that are greater than 90

Index	Member
1	[[Zachary,75,81, ...],[Sophia,74,86, ...],[Christopher,71,...
2	[[Natalie,84,90, ...],[Jessica,87,88, ...],[Brianna,89,90, ...
3	[[Sean,98,86, ...]]

Group that are greater than 80

Index	Name	Math	Chinese	English
1	Natalie	84	90	84
2	Jessica	87	88	78
3	Brianna	89	90	75
4	Emma	88	84	94
5	Hannah	90	76	95
6	Sean	98	86	81
7	Tyler	87	78	93

SALARY field contains missing values. Use random values in this field to fill in missing values (conventional method)

	A	B
1	=file("E:/txt/salary_nan.txt").import@t(EID,NAME,GENDER,SALARY)	
2	=A1.select(!SALARY)	/Grouped according to Boolean values, the missing values and the non-missing values are divided into two groups
3	=A1\A2	/Take out SALARY values that are not missing values
4	>A2.field(-1,A3.(SALARY)(rand(A3.len()+1)))	/Filling missing values with random values in SALARY

A1 when A1 executed

Index	EID	NAME	GENDER	SALARY
57	57	Megan	F	(null)
58	58	Andrew	M	13000
59	59	David	M	5000
60	60	Taylor	F	12000
61	61	Joseph	M	5000
62	62	Emily	F	6500
63	63	Samantha	F	7000
64	64	Nathan	M	5000
65	65	Michael	M	8000
66	66	Madison	F	6500
67	67	Cameron	M	5000
68	68	Ashley	F	10000
69	69	Brandon	M	7000
70	70	Brandon	M	10000
71	71	Tyler	M	5000
72	72	Madison	F	(null)
73	73	Justin	M	10000

A1 after A4 executed

Index	EID	NAME	GENDER	SALARY
57	57	Megan	F	8000
58	58	Andrew	M	13000
59	59	David	M	5000
60	60	Taylor	F	12000
61	61	Joseph	M	5000
62	62	Emily	F	6500
63	63	Samantha	F	7000
64	64	Nathan	M	5000
65	65	Michael	M	8000
66	66	Madison	F	6500
67	67	Cameron	M	5000
68	68	Ashley	F	10000
69	69	Brandon	M	7000
70	70	Brandon	M	10000
71	71	Tyler	M	5000
72	72	Madison	F	8000
73	73	Justin	M	10000

A2 when A2 executed

Index	EID	NAME	GENDER	SALARY
1	57	Megan	F	(null)
2	72	Madison	F	(null)
3	96	Jasmine	F	(null)
4	166	Emily	F	(null)
5	195	Michael	M	(null)

A2 after A4 executed

Index	EID	NAME	GENDER	SALARY
1	57	Megan	F	8000
2	72	Madison	F	8000
3	96	Jasmine	F	8000
4	166	Emily	F	8000
5	195	Michael	M	8000

A3

Index	Member
1	7000
2	11000
3	9000
4	7000
5	16000
6	11000
7	9000
8	11000
9	3000
10	13000

SALARY field contains missing values. Use random values in this field to fill in missing values (Efficient method)

	A	B
1	=file("E:/txt/salary_nan.txt").import@t(EID,NAME,GENDER,SALARY)	
2	=A1.group(!SALARY)	//Grouped according to Boolean values, the missing values and the non-missing values are divided into two groups
3	=A2(1).(SALARY)	/Take out SALARY values that are not missing values
4	>A2(2).run(~.field(-1,A3(rand(A3.len()+1))))	/Filling missing values with random values in SALARY

A1 when A1 executed

Index	EID	NAME	GENDER	SALARY
57	57	Megan	F	(null)
58	58	Andrew	M	13000
59	59	David	M	5000
60	60	Taylor	F	12000
61	61	Joseph	M	5000
62	62	Emily	F	6500
63	63	Samantha	F	7000
64	64	Nathan	M	5000
65	65	Michael	M	8000
66	66	Madison	F	6500
67	67	Cameron	M	5000
68	68	Ashley	F	10000
69	69	Brandon	M	7000
70	70	Brandon	M	10000
71	71	Tyler	M	5000
72	72	Madison	F	(null)
73	73	Justin	M	10000

A1 after A4 executed

Index	EID	NAME	GENDER	SALARY
57	57	Megan	F	10000
58	58	Andrew	M	13000
59	59	David	M	5000
60	60	Taylor	F	12000
61	61	Joseph	M	5000
62	62	Emily	F	6500
63	63	Samantha	F	7000
64	64	Nathan	M	5000
65	65	Michael	M	8000
66	66	Madison	F	6500
67	67	Cameron	M	5000
68	68	Ashley	F	10000
69	69	Brandon	M	7000
70	70	Brandon	M	10000
71	71	Tyler	M	5000
72	72	Madison	F	10000
73	73	Justin	M	10000

A2

Index	Member
1	[[1,Rebecca,F, ...],[2,Ashley,F, ...],[3,Rachel,F, ...], ...]
2	[[57,Megan,F, ...],[72,Madison,F, ...],[96,Jasmine,F, ...]...

A2(2) when A2 executed

Index	EID	NAME	GENDER	SALARY
1	57	Megan	F	(null)
2	72	Madison	F	(null)
3	96	Jasmine	F	(null)
4	166	Emily	F	(null)
5	195	Michael	M	(null)

A2(2) after A4 executed

Index	EID	NAME	GENDER	SALARY
1	57	Megan	F	10000
2	72	Madison	F	10000
3	96	Jasmine	F	16000
4	166	Emily	F	7000
5	195	Michael	M	7000

A3

Index	Member
1	7000
2	11000
3	9000
4	7000
5	16000
6	11000
7	9000
8	11000
9	3000
10	13000

Of course, align and enum can also be used to improve efficiency while ensuring that records exist fulfilling the statistics conditions.

Example: Calculate the average age of employees whose salaries exceed 10,000 and employees whose salaries are less than 10,000.

	A	
1	=demo.query("select EID,NAME,GENDER,BIRTHDAY,SALARY from EMPLOYEE")	
2	[false,true]	/[?<10000,?>=10000]
3	=A1.align@a(A2,SALARY>=10000)	/=A1.enum(A6,SALARY)
4	=A3.new(A2(#):wether_gt_100000,~.avg(age(BIRTHDAY)):avg_age)	

A1

Index	EID	NAME	GENDER	BIRTHDAY	SALARY
1	1	Rebecca	F	1974-11-20	7000
2	2	Ashley	F	1980-07-19	11000
3	3	Rachel	F	1970-12-17	9000
4	4	Emily	F	1985-03-07	7000
5	5	Ashley	F	1975-05-13	16000
6	6	Matthew	M	1984-07-07	11000
7	7	Alexis	F	1972-08-16	9000
8	8	Megan	F	1979-04-19	11000
9	9	Victoria	F	1983-12-07	3000
10	10	Ryan	M	1976-03-12	13000

A3

Index	Member
1	[[1,Rebecca,F, ...],[3,Rachel,F, ...],[4,Emily,F, ...], ...]
2	[[2,Ashley,F, ...],[5,Ashley,F, ...],[6,Matthew,M, ...], ...]

Index	EID	NAME	GENDER	BIRTHDAY	SALARY
1	1	Rebecca	F	1974-11-20	7000
2	3	Rachel	F	1970-12-17	9000
3	4	Emily	F	1985-03-07	7000
4	7	Alexis	F	1972-08-16	9000
5	9	Victoria	F	1983-12-07	3000

A4

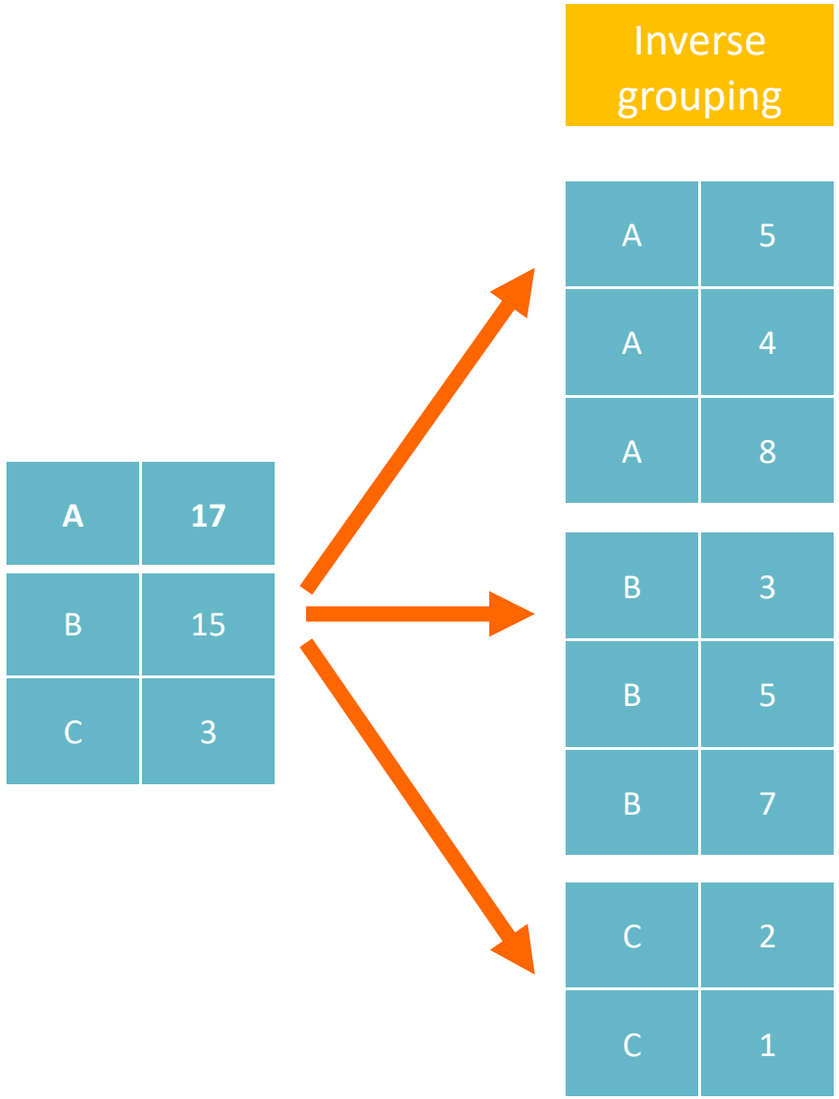
Index	wether_gt_100000	avg_age
1	false	40.60052219321149
2	true	40.888888888888886

05

Inverse grouping

The essence of grouping is the process of splitting and then applying.

But we often encounter the inverse operation of splitting a record into multiple pieces, i.e. Inverse grouping.



Example: List the detailed product number

	A
1	=file("E:/txt/product.txt").import@t()
2	=A1.conj(to(beginid,endid).new(productname:product,~:productid))
3	=A1.news(to(beginid,endid);productname:product,~:productid)

A1~A3 results:

Index	productname	beginid	endid
1	product_a	201	207
2	product_b	151	155
3	product_c	99	131
4	product_d	476	487
5	product_e	77	88

Index	product	productid
1	product_a	201
2	product_a	202
3	product_a	203
4	product_a	204
5	product_a	205
6	product_a	206
7	product_a	207
8	product_b	151
9	product_b	152
10	product_b	153
11	product_b	154
12	product_b	155
13	product_c	99
14	product_c	100
15	product_c	101
16	product_c	102

=

Index	product	productid
1	product_a	201
2	product_a	202
3	product_a	203
4	product_a	204
5	product_a	205
6	product_a	206
7	product_a	207
8	product_b	151
9	product_b	152
10	product_b	153
11	product_b	154
12	product_b	155
13	product_c	99
14	product_c	100
15	product_c	101
16	product_c	102

Looking back at the example of “**calculating each person's starting and ending duty time**”, can we use news to restore the duty schedule?

	A
1	=file("E:/txt/duty_r.txt").import@t()
2	=A1.news(if(begin==end,[begin],periods(begin,end));name,~:date)

A1, A2 results:

Index	name	begin	end
1	Emily	2018-03-01	2018-03-03
2	Johnson	2018-03-04	2018-03-04
3	Ashley	2018-03-05	2018-03-05
4	Emily	2018-03-06	2018-03-07
5	Ashley	2018-03-08	2018-03-08
6	Emily	2018-03-09	2018-03-09
7	Ashley	2018-03-10	2018-03-10
8	Johnson	2018-03-11	2018-03-15
9	Ashley	2018-03-16	2018-03-16
10	Johnson	2018-03-17	2018-03-17

Index	name	date
1	Emily	2018-03-01
2	Emily	2018-03-02
3	Emily	2018-03-03
4	Johnson	2018-03-04
5	Ashley	2018-03-05
6	Emily	2018-03-06
7	Emily	2018-03-07
8	Ashley	2018-03-08
9	Emily	2018-03-09
10	Ashley	2018-03-10

THANKS