

SPL WIN

SPL WIN Course

- ✓ Useful for zero foundation coding newbies
- ✓ Great help to promotion and pay raise
- ✓ Super practical office skill
- ✓ Secretly-becoming-excellent technique



✦ CONTENTS



Chapter 1 Functional sections

- 1.1 Data files
- 1.2 Dataset panel
- 1.3 History command
- 1.4 Edit command

Chapter 2 Interface computations

- 2.1 Add computed columns
- 2.2 Filter data
- 2.3 Sort data
- 2.4 Group data
- 2.5 Concatenate detailed data
- 2.6 Select fields
- 2.7 Transpose rows/columns to columns/rows
- 2.8 Data association
- 2.9 Set operations

Chapter 3 Comprehensive scenarios

- 3.1 Aggregation on a computed column in a grouped table
- 3.2 Intragroup sorting on a grouped table
- 3.3 Intragroup filtering on a grouped table
- 3.4 Retain the first row of each group
- 3.5 Grouping & aggregation after filtering
- 3.6 Compute link relative in a multilayer-structure table
- 3.7 Comparison between multiple tables

Chapter 4 Comprehensive use cases

- 4.1 Get stock transaction records meeting the specified condition
- 4.2 Compute LRR and YOY based on car sales table
- 4.3 Associate worksheets and compute salaries
- 4.4 Find salespeople who rank top5 every month

Chapter 5 Editing result set

- 5.1 Data rows
- 5.2 Data columns
- 5.3 Cell value expression
- 5.4 Save and export

SPL WIN Course

Chapter 1

Functional sections

SPL WIN



◆ Functional sections



Below is the SPL WIN interface after the application is started. The interface consists of 4 sections:

1. Data files
2. Dataset panel
3. History command
4. Edit command

The screenshot shows the SPL WIN interface with four sections highlighted in blue:

- 1. Data files:** A file explorer showing the directory structure. The 'data' folder is expanded, showing files like 'orders.1.1.xlsx', 'scores.1.1.txt', and 'student.1.1.txt'.
- 2. Dataset panel:** A table displaying the contents of the selected file. The table has columns: OrderID, CustomerID, EmployeeID, OrderDate, and ShipCity. The data rows are:

| | OrderID | CustomerID | EmployeeID | OrderDate | ShipCity |
|---|-----------|------------|------------|-----------|-------------|
| 1 | 202003050 | LA3235 | 501 | 3/5/2020 | Los Angeles |
| 2 | 202005140 | NY7866 | 500 | 5/14/2020 | New York |
| 3 | 202005203 | CH0987 | 356 | 5/20/2020 | Chicago |
| 4 | 202006075 | DT8880 | 390 | 7/7/2020 | Detroit |
| 5 | 202006302 | WT2204 | 400 | 6/30/2020 | Washington |
- 3. History command:** A text area showing the command history. The commands listed are:

```
result.run(Name = upper(Name))
T("scores.1.1.txt")
T("student.1.1.txt")
```
- 4. Edit command:** A text area for editing the current command. The command shown is:

```
T("student.1.1.txt")
```

The boxes highlighted in blue are indicators of interface sections

CONTENTS



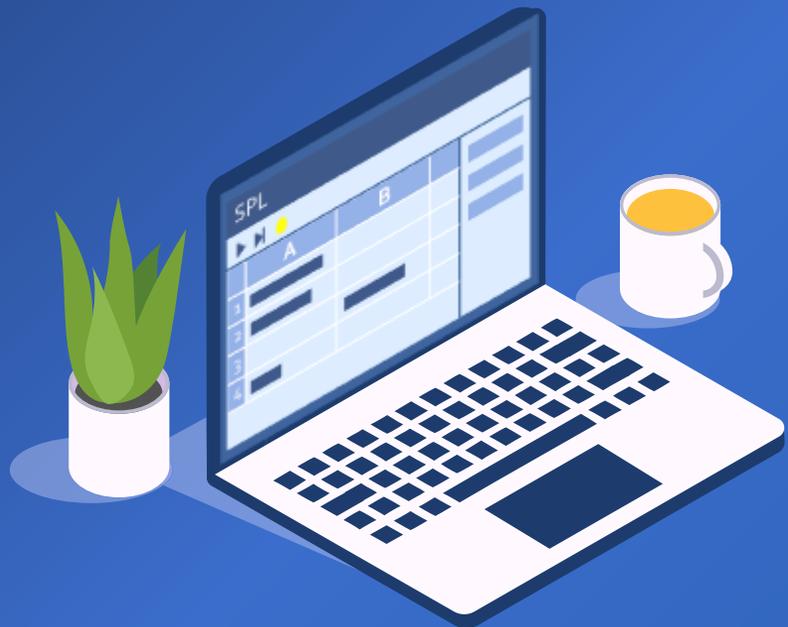
1.1 Data files

1.2 Dataset panel

1.3 History command

1.4 Edit command

CONTENTS



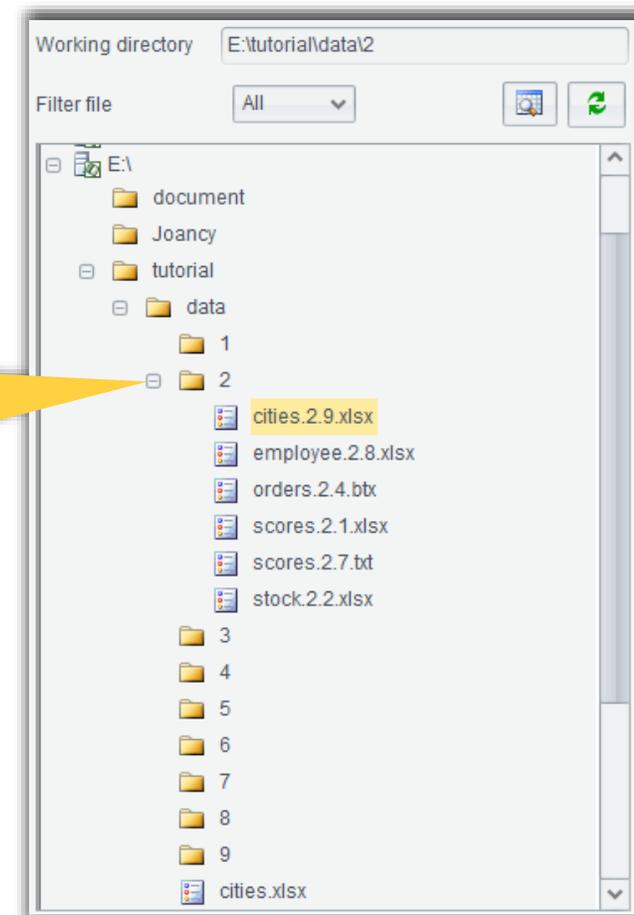
1.1 Data files

✦ 1.1 Data files



Computable file formats include text files (txt and csv), Excel files, and bin files (btx, which is the esProc proprietary binary format):

As there are too many files in the directory, the file tree adopts lazy loading – files in a folder will be listed only when it is selected



✦ 1.1 Data files



Double-click a file name and the corresponding data table is opened according to the default rule (such as displaying the 1st row as field names and using Tab to separate the text file):

Filter file: All

Data files

- C:\
- D:\
- E:\
 - document
 - Joancy
 - tutorial
 - data
 - 1
 - emp.1.1.xlsx
 - orders.1.1.xlsx
 - scores.1.1.txt
 - student.1.1.txt
 - 2
 - 3
 - 4
 - F:\
 - G:\

student

Row count: 5

| | ID | Name | Gender |
|---|----|-------|--------|
| 1 | 10 | Mike | M |
| 2 | 27 | Rose | F |
| 3 | 3 | Susan | F |
| 4 | 40 | Tom | M |
| 5 | 5 | Smith | M |

In this crash course, content highlighted by a red circle/oval means a mouse click action

By default, table name will be directly displayed as the corresponding file name

✦ 1.1 Data files



If format of a file does not conform to the default open rule, error happens when it is opened directly, such as **scores.1.1.txt** :

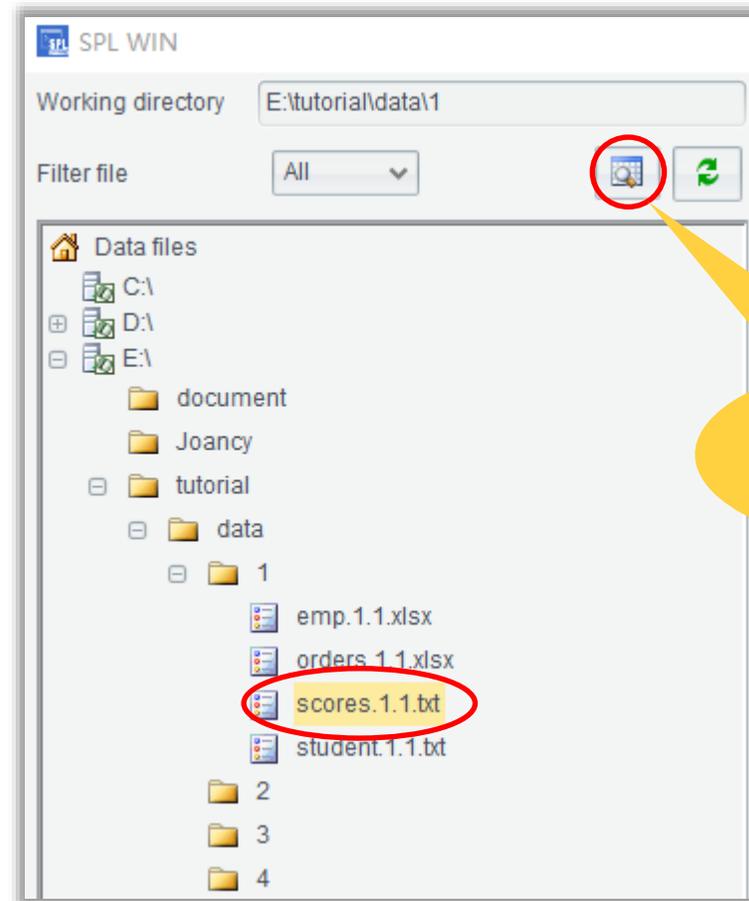
| | names | LastNames | Subject | Score | Birthday |
|----|---------------------------------------|-----------|---------|-------|----------|
| 1 | Pascal,Seguin,Math,59,2001/1/1 | | | | |
| 2 | Pascal,Seguin,Science,48,2001/1/1 | | | | |
| 3 | Christophe,Stenac,Math,77,2001/1/1 | | | | |
| 4 | Christophe,Stenac,Science,70,2001/1/1 | | | | |
| 5 | Loic,Teuliere,Math,36,2001/1/1 | | | | |
| 6 | Loic,Teuliere,Science,54,2001/1/1 | | | | |
| 7 | Christophe,Pracht,Math,25,2001/1/1 | | | | |
| 8 | Christophe,Pracht,Science,56,2001/1/1 | | | | |
| 9 | Florian,Puig,Math,66,2001/1/1 | | | | |
| 10 | Florian,Puig,Science,75,2001/1/1 | | | | |
| 11 | Eric,Self,Math,32,2001/1/1 | | | | |
| 12 | Eric,Self,Science,42,2001/1/1 | | | | |
| 13 | Benjamin,Shepherd,Math,37,2001/1/1 | | | | |
| 14 | Benjamin,Shepherd,Science,97,2001/1/1 | | | | |
| 15 | Emmanuel,Schauly,Math,56,2001/1/1 | | | | |

Incorrect field splitting; all columns are displayed in one cell

✦ 1.1 Data files



To deal with the error, open the File dialog and set up Open options in a more detailed way:



Select a file
and click the
Browse button

✦ 1.1 Data files



On the interface, adjust the delimiter to a correct one:

Text file

Name: scores Result type: Table

File name: E:\tutorial\data\1\scores.1.1.txt

Charset: Default Delimiter: TAB Refresh

Segment NO: Segment count:

Import fields

| Index | Field | Type | Format | Select |
|-------|----------------------------------|---------|--------|-------------------------------------|
| 1 | FirstNames,LastNames,Subject,... | Default | | <input checked="" type="checkbox"/> |

Import the first row as field names
 Remove quotation marks
 Double quotation marks as escape character
 Retrieve data in multiple threads
 Strip leading and trailing spaces
 Report error when column count does not match that at row 1

Load fields incorrectly with default options

Select the correct comma from the drop-down list, or directly enter the English comma, and click the Refresh button

◆ 1.1 Data files



In the File dialog, we can choose to import some of the fields:

Text file dialog settings:

- Name: scores
- Result type: Table
- File name: E:\tutorial\data1\scores.1.1.txt
- Charset: Default
- Delimiter: ,
- Segment NO:
- Segment count:

Import fields table:

| Index | Field | Type | Format | Select |
|-------|------------|---------|--------|-------------------------------------|
| 1 | FirstNames | Default | | <input checked="" type="checkbox"/> |
| 2 | LastNames | Default | | <input checked="" type="checkbox"/> |
| 3 | Subject | Default | | <input checked="" type="checkbox"/> |
| 4 | Score | Default | | <input checked="" type="checkbox"/> |
| 5 | Birthday | Default | | <input type="checkbox"/> |

Preview window 'scores' (Row count: 16):

| | FirstNames | Last | | |
|----|------------|----------|---------|----|
| 1 | Pascal | Seguin | | |
| 2 | Pascal | Seguin | | |
| 3 | Christophe | Stenac | Math | 87 |
| 4 | Christophe | Stenac | Science | 70 |
| 5 | Loic | Teuliere | Math | 36 |
| 6 | Loic | Teuliere | Science | 54 |
| 7 | Christophe | Pracht | Math | 25 |
| 8 | Christophe | Pracht | Science | 56 |
| 9 | Florian | Puig | Math | 66 |
| 10 | Florian | Puig | Science | 75 |
| 11 | Eric | Self | Math | 32 |
| 12 | Eric | Self | Science | 42 |
| 13 | Benjamin | Shepherd | Math | 37 |
| 14 | Benjamin | Shepherd | Science | 97 |
| 15 | Emmanuel | Schauly | Math | 56 |

Callout 1: Import a data table without Birthday field

Callout 2: Click the checkbox to disable importing the Birthday field

Callout 3: After the new comma delimiter is set up and Refresh is performed, all fields are listed

◆ 1.1 Data files



In the File dialog, we can set up the type of big file processing result as the cursor:

Text file

Name: student Result type: **Cursor**

File name: E:\tutorial\data\1\student.1.1.txt

Charset: Default Delimiter: TAB

Segment NO: Segment count:

| Index | Field | Type | Format | Select |
|-------|--------|---------|--------|-------------------------------------|
| 1 | ID | Default | | <input checked="" type="checkbox"/> |
| 2 | Name | Default | | <input checked="" type="checkbox"/> |
| 3 | Gender | Default | | <input checked="" type="checkbox"/> |

Import the first row as field names

Use double quotation marks as escape character

Keep leading and trailing spaces

Remove quotation marks

Retrieve data in multiple threads

Report error when column count does not match that at row 1

scores student

Rows to fetch: 1024 Row count: 5

| | ID | Name | Gender |
|---|----|-------|--------|
| 1 | 10 | Mike | M |
| 2 | 27 | Rose | F |
| 3 | 3 | Susan | F |
| 4 | 10 | Tom | M |
| 5 | | Smith | M |

Highlight cursor in red to distinguish it from the table sequence; we can set up the number of rows to be fetched from a cursor

◆ 1.1 Data files



Data types of text files, default numbers and characters can be all parsed correctly:

Text file dialog details:

- Name: scores
- Result type: Table
- File name: E:\tutorial\data1\scores.1.1.txt
- Charset: Default
- Delimiter: ,
- Segment NO:
- Segment count:

| Index | Field | Type | Format | Select |
|-------|------------|---------|--------|-------------------------------------|
| 1 | FirstNames | Default | | <input checked="" type="checkbox"/> |
| 2 | LastNames | Default | | <input checked="" type="checkbox"/> |
| 3 | Subject | Default | | <input checked="" type="checkbox"/> |
| 4 | Score | Default | | <input checked="" type="checkbox"/> |
| 5 | Birthday | Default | | <input checked="" type="checkbox"/> |

Preview window data (rows 1-14):

| | FirstNames | LastNames | Subject | Score | Birthday |
|----|------------|-----------|---------|-------|----------|
| 1 | Pascal | Seguin | Math | 59 | 2001/1/1 |
| 2 | Pascal | Seguin | Science | 48 | 2001/1/1 |
| 3 | Christophe | Stenac | Math | 87 | 2001/1/1 |
| 4 | Christophe | Stenac | Science | 70 | 2001/1/1 |
| 5 | Loic | Teuliere | Math | 36 | 2001/1/1 |
| 6 | Loic | Teuliere | Science | 54 | 2001/1/1 |
| 7 | Christophe | Pracht | Math | 25 | 2001/1/1 |
| 8 | Christophe | Pracht | Science | 56 | 2001/1/1 |
| 9 | | | | | 2001/1/1 |
| | | | | 75 | 2001/1/1 |
| | | | | 32 | 2001/1/1 |
| | | | | 42 | 2001/1/1 |
| | | | | 37 | 2001/1/1 |
| 14 | | | Science | 97 | 2001/1/1 |

The date type is special, and maybe the parsing is incorrect

✦ 1.1 Data files



Set up default parse format for date type data through esProc options:

The screenshot shows the 'SPL Options' dialog box with the 'Appearance' tab selected. The 'Date format' dropdown is set to 'yyyy-MM-dd' and the 'Time format' dropdown is set to 'HH:mm:ss'. Both dropdowns are highlighted with a blue box. Other settings include 'Log file name' (E:/test.log), 'Search path' (demo), 'Main path' (D:\), 'Temp path', 'Initialization program', 'External library directory', 'Custom function file' (customFunctions.properties), 'Date time format' (yyyy-MM-dd HH:mm:ss), 'Parallel limit' (900), 'File buffer(Byte)' (65536), 'Composite table block size (Byte)' (1048576), 'Default charset name' (GBK), 'Default number of multiple cursors' (1), 'Missing values (Comma-separated)' (nan,null,n/a), and 'Cursor fetch count' (9999). Buttons for 'Browse', 'OK', and 'Cancel' are visible.

By default, dates in the text can be correctly parsed only when their formats are consistent with the option settings

◆ 1.1 Data files



If date formats in the text are not consistent with the option settings and you are reluctant to modify the settings:

Text file

Name: scores Result type: Table

File name: E:\tutorial\data\1\scores.1.1.txt

Charset: Default Delimiter: ,

Segment NO: Segment count:

Import fields

| Index | Field | Type | Format |
|-------|------------|---------|------------|
| 1 | FirstNames | Default | |
| 2 | LastNames | Default | |
| 3 | Subject | Default | |
| 4 | Score | Default | |
| 5 | Birthday | Date | yyyy/MM/dd |

Import the first row as field names Remove quotation marks
 Use double quotation marks as escape character Retrieve data in multiple threads
 Keep leading and trailing spaces Report error when column count does not match that at row 1

◆ 1.1 Data files



Open the other Excel sheets from the File dialog:

Working directory: E:\tutorial\data1

Filter file: All

Data files

- emp.1.1.xlsx
- orders.1.1.xlsx**
- scores.1.1.txt
- student.1.1.txt
- 2
- 3
- 4
- F:\
- G:\

Excel file

Name: DETAIL Result type: Table

File name: E:\tutorial\data1\orders.1.1.xlsx

Begin row: End row:

Sheet: **DETAIL** Password:

Import fields

| Index | Field |
|-------|-----------|
| 1 | ProductID |
| 2 | UnitPrice |
| 3 | OrderID |
| 4 | Quantity |

Import the first row as field names Ignore empty rows

Trim all

Select **orders.1.1.xlsx** and click Browse button

orders.1.1.xlsx contains two sheets – ORDER and DETAIL

◆ 1.1 Data files



When an Excel file is encrypted through a password, you need to enter the password and click Refresh to list the fields:

Working directory: E:\tutorial\data1

Filter file: All

Data files:

- C:\
- D:\
- F:\
- G:\

Excel file:

Name: emp

Result type: Table

File name: E:\tutorial\data1\emp.1.1.xlsx

Begin row:

End row:

Sheet:

Password:

Import fields:

| Index | Field | Select |
|-------|-------|--------|
|-------|-------|--------|

Import the first row as field names

Ignore empty rows

Trim all

Select **emp.1.1.xlsx** and click Browse button

As the file is encrypted with the password, no sheets and fields are displayed

◆ 1.1 Data files



Enter the password correctly and click Refresh button to display sheets and fields:

The image shows two overlapping windows. The left window is an 'Excel file' import dialog. It has fields for 'Name' (emp), 'File name' (E:\tutorial\data\1\emp.1.1.xlsx), 'Sheet' (s1), and 'Password' (masked with five dots). A red box highlights the password field, and a yellow callout bubble points to it with the text: 'Enter the file's password (12345) and click Refresh button'. The 'Refresh' button is also circled in red. Below these fields is a table of fields to import, with all 'Select' checkboxes checked.

| Index | Field | Select |
|-------|----------|-------------------------------------|
| 1 | EID | <input checked="" type="checkbox"/> |
| 2 | NAME | <input checked="" type="checkbox"/> |
| 3 | SURNAME | <input checked="" type="checkbox"/> |
| 4 | GENDER | <input checked="" type="checkbox"/> |
| 5 | STATE | <input checked="" type="checkbox"/> |
| 6 | BIRTHDAY | <input checked="" type="checkbox"/> |
| 7 | HIREDATE | <input checked="" type="checkbox"/> |
| 8 | DEPT | <input checked="" type="checkbox"/> |

The right window shows the 'emp' table with the following data:

| | EID | NAME | SURNAME | GENDER | STATE | BIRTH |
|----|-----|----------|---------|--------|--------------|--------|
| 1 | 1 | Rebecca | Moore | F | California | 1974- |
| 2 | 2 | Ashley | Wilson | F | New York | 1980-0 |
| 3 | 3 | Rachel | Johnson | F | New Mexico | 1970- |
| 4 | 4 | Emily | Smith | F | Texas | 1985- |
| 5 | 5 | Ashley | Smith | F | Texas | 1975-0 |
| 6 | 6 | Matthew | Johnson | M | California | 1984-0 |
| 7 | 7 | Alexis | Smith | F | Illinois | 1972-0 |
| 8 | 8 | Megan | Wilson | F | California | 1979-0 |
| 9 | 9 | Victoria | Davis | F | Texas | 1983- |
| 10 | 10 | Ryan | Johnson | M | Pennsylvania | 1976-0 |
| 11 | 11 | Jacob | Moore | M | Texas | 1974- |
| 12 | 12 | Jessica | Davis | F | New York | 1980-0 |

CONTENTS



✦ 1.2 Dataset panel



View and edit data in an opened data table, and perform computations on it. Icons on the toolbar let you compute a data table as if on a calculator:

Working directory: E:\tutorial\data\1

Filter file: All

Data files:

- C:\
- D:\
- E:\
 - document
 - Joancy
 - tutorial
 - data
 - 1
 - orders.1.1.xlsx
 - scores.1.1.txt
 - student.1.1.txt

Result: orders

| | OrderID | CustomerID | EmployeeID | OrderDate | ShipCity |
|---|-----------|------------|------------|-----------|-------------|
| 1 | 202003050 | LA3235 | 501 | 3/5/2020 | Los Angeles |
| 2 | 202005140 | NY7866 | 200 | 5/14/2020 | New York |
| 3 | 202005203 | CH0987 | 300 | 5/20/2020 | Chicago |
| 4 | 202006075 | DT8880 | 390 | 6/7/2020 | Detroit |
| 5 | 202006302 | WT2204 | 438 | 6/10/2020 | Washington |

Computation toolbar

◆ 1.2 Dataset panel



A result returned from executing a SPL command at the command line can only be viewed and won't be named; an existing result set will be overwritten by the result of another execution from the command line:

The screenshot shows the SPL WIN interface. The working directory is E:\tutorial\data\1. The filter file is set to All. The Data files pane shows a tree view with folders C:\, D:\, and E:\. Under E:\, there are folders document, Joancy, and tutorial. Under tutorial, there is a folder data, which contains a folder 1. Inside folder 1, there are files orders.1.1.xlsx, scores.1.1.txt, and student.1.1.txt. The Result pane shows a table with columns Index, ID, Name, and Gender. The table contains 5 rows of data. The History command pane shows the command T("student.1.1.txt") being executed. A yellow callout bubble points to the 'orders' result set and the 'Run' button (a blue play icon) in the History command pane. The callout bubble contains the text: "Result set of execution from the command line does not have a name".

| Index | ID | Name | Gender |
|-------|----|-------|--------|
| 1 | 10 | Mike | M |
| 2 | 27 | Rose | F |
| 3 | 3 | Susan | F |
| 4 | 40 | Tom | M |
| 5 | 5 | Smith | M |

Result set of execution from the command line does not have a name

◆ 1.2 Dataset panel



To retain the result data table of the current computation for use in subsequent computations, it should be given a name:

A screenshot of a data table interface. The table has columns 'Index', 'ID', 'Name', and 'Gender'. The data rows are: (1, 10, Mike, M), (2, 27, Rose, F), (3, 3, Susan, F), (4, 40, Tom, M), (5, 5, Smith, M). A red circle highlights a pencil icon in the top toolbar of the 'Result' panel.

| Index | ID | Name | Gender |
|-------|----|-------|--------|
| 1 | 10 | Mike | M |
| 2 | 27 | Rose | F |
| 3 | 3 | Susan | F |
| 4 | 40 | Tom | M |
| 5 | 5 | Smith | M |

Click the icon to **rename** it

An 'Input' dialog box with a question mark icon and the text 'Please enter a new name'. The text input field contains the word 'result'. There are 'OK' and 'Cancel' buttons at the bottom.

Note: The name must be a legal identifier

A screenshot of a data table interface showing a computation toolbar above the table. The toolbar includes icons for 'ABC 123', a filter, a sort (A-Z), a refresh, a save, a link, and a refresh. The table data is the same as in the previous screenshot.

| | ID | Name | Gender |
|---|----|-------|--------|
| 1 | 10 | Mike | M |
| 2 | 27 | Rose | F |
| 3 | 3 | Susan | F |
| 4 | 40 | Tom | M |
| 5 | 5 | Smith | M |

The computation tool bar only displays for a named data table

Exception: Computation tool bar won't be displayed if the result is a sequence

✦ 1.2 Dataset panel



Open `orders.2.4.btx`, we can see that fields are displayed in different colors according to their own types:

| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 1 | UJRN | 17 | 392.0 | 2012-11-02 15:28:05 |
| 2 | 2 | SJCH | 6 | 4802.0 | 2012-11-09 15:28:05 |
| 3 | 3 | UJRN | 16 | 13500.0 | 2012-11-05 15:28:05 |
| 4 | 4 | PWQ | 9 | 26100.0 | 2012-11-08 15:28:05 |
| 5 | 5 | PWQ | 11 | 4410.0 | 2012-11-12 15:28:05 |
| 6 | 6 | HANAR | 18 | 6174.0 | 2012-11-07 15:28:05 |
| 7 | 7 | EGU | 2 | 17800.0 | 2012-11-06 15:28:05 |
| 8 | 8 | VILJX | 7 | 2156.0 | 2012-11-09 15:28:05 |
| 9 | 9 | JAYB | 14 | 17400.0 | 2012-11-12 15:28:05 |
| 10 | 10 | JAXE | 19 | 19200.0 | 2012-11-12 15:28:05 |
| 11 | 11 | SJCH | 7 | 13700.0 | 2012-11-10 15:28:05 |
| 12 | 12 | QUICK | 11 | 21200.0 | 2012-11-13 15:28:05 |
| 13 | 13 | HL | 12 | 21400.0 | 2012-11-21 15:28:05 |
| 14 | 14 | JAYB | 1 | 7644.0 | 2012-11-16 15:28:05 |

Integers are displayed in blue and right-aligned

Floating point numbers are displayed in light red and right-aligned

Characters and dates are displayed in black

CONTENTS



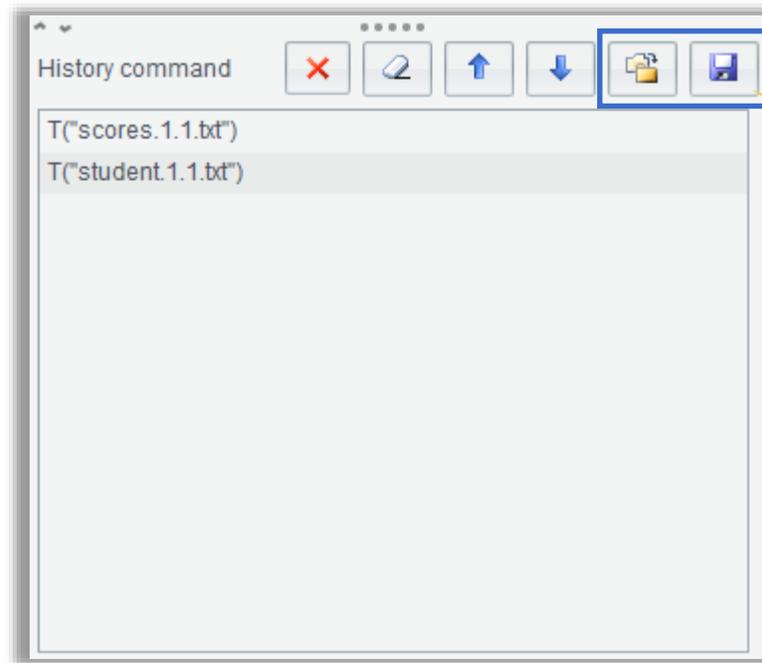
History command

✦ 1.3 History command



Commands executed from the command line will be automatically appended to the “History command” section.

The history commands will be automatically stored and loaded:



Save the history command list as an external file, or load it from the external

CONTENTS



✦ 1.4 Edit command



SPL has a rich collection of function libraries. The toolbar on “Dataset panel” only implements some of the commonly used functions. To use more complicated functions, we can manually code and execute it in “Edit command” section:

The screenshot shows the SPL interface. At the top, there's a "Result" panel with a toolbar. Below it, a table displays data with columns "Index", "ID", "Name", and "Gender". The table content is as follows:

| Index | ID | Name | Gender |
|-------|----|-------|--------|
| 1 | 10 | MIKE | M |
| 2 | 27 | ROSE | F |
| 3 | 3 | SUSAN | F |
| 4 | 40 | TOM | M |
| 5 | 5 | SMITH | M |

Below the table is the "Edit command" section, which includes a toolbar with a refresh icon and a play icon, and a text area containing the command: `result.run(Name = upper(Name))`. To the right of the text area is a checkbox labeled "Enable prompt" which is checked.

Code the run() function in “Edit command” section to convert names to uppercase

Check “Enable prompt” to type in function parameters quickly and efficiently

SPL WIN Course

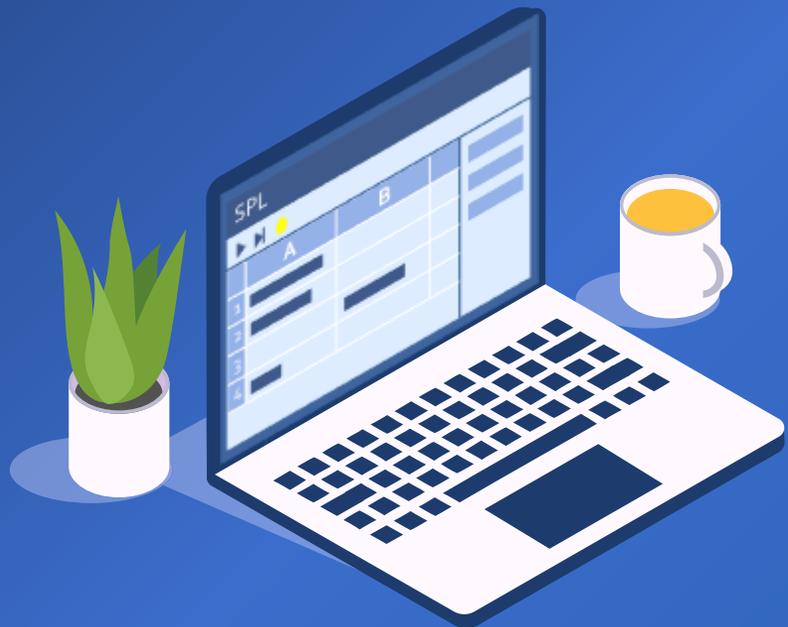
Chapter 2

Interface computations

SPL WIN



CONTENTS



2.1 Add computed columns

2.2 Filter data

2.3 Sort data

2.4 Group data

CONTENTS



2.5 Concatenate detailed data

2.6 Select fields

2.7 Transpose rows/columns to columns/rows

2.8 Data association

2.9 Set operations

CONTENTS



Add computed columns

✦ 2.1 Add computed columns



Below is Excel table **scores.2.1.xlsx** recording students subject scores. We need to compute total score (Total) of each student and the average score (Average:

| | A | B | C | D | E | F |
|----|-----------|-----------|---------|------|----|---|
| 1 | CLASS | STUDENTID | English | Math | PE | |
| 2 | Class one | 1 | 84 | 77 | 69 | |
| 3 | Class one | 2 | 81 | 80 | 97 | |
| 4 | Class one | 3 | 75 | 86 | 67 | |
| 5 | Class one | 4 | 96 | 63 | 81 | |
| 6 | Class one | 5 | 72 | 60 | 91 | |
| 7 | Class one | 6 | 90 | 69 | 72 | |
| 8 | Class one | 7 | 75 | 90 | 60 | |
| 9 | Class one | 8 | 67 | 86 | 51 | |
| 10 | Class one | 9 | 93 | 73 | 83 | |
| 11 | Class one | 10 | 52 | 97 | 84 | |
| 12 | Class one | 11 | 78 | 63 | 79 | |
| 13 | Class one | 12 | 65 | 71 | 79 | |
| 14 | Class one | 13 | 61 | 97 | 59 | |
| 15 | Class one | 14 | 52 | 60 | 86 | |

✦ 2.1 Add computed columns



Method 1: Use dialog to configure the computation

Double-click **scores.2.1.xlsx** in data/2 to open the data table whose default name is scores:

Default table name;
if there is a
namesake table,
add an serial
number after it

Click **Add
computed
columns** icon

| | CLASS | STUDENTID | English | Math | PE |
|----|-----------|-----------|---------|------|----|
| 1 | Class one | 1 | 84 | 77 | 69 |
| 2 | Class one | 2 | 81 | 80 | 97 |
| 3 | Class one | 3 | 75 | 86 | 67 |
| 4 | Class one | 4 | 96 | 63 | 81 |
| 5 | Class one | 5 | 72 | 60 | 91 |
| 6 | Class one | 6 | 90 | 69 | 72 |
| 7 | Class one | 7 | 75 | 90 | 60 |
| 8 | Class one | 8 | 67 | 86 | 51 |
| 9 | Class one | 9 | 93 | 73 | 83 |
| 10 | Class one | 10 | 52 | 97 | 84 |
| 11 | Class one | 11 | 78 | 63 | 79 |

✦ 2.1 Add computed columns



On the “Computed columns” interface, add expressions and aliases for the computed columns (as shown below).

It is convenient to do the editing on the interface. We can add multiple computed columns at one time.

Computed column

Name: scores Source table: scores

Options: [Empty field]

Computed column: [+] [-] [↑] [↓]

| Index | Expression | Alias |
|-------|------------------|---------|
| 1 | English+Math+PE | Total |
| 2 | round(Total/3,2) | Average |

Source fields (Double click to select): CLASS, English, PE, STUDENTI

Parallel computation

OK Cancel

It is the table name after the computation is finished. If it is namesake with the “Source table” on the right, it will overwrite the table currently-computed table “Name”

The red square in this course represents content that by default does not exist in the interface and needs editing

✦ 2.1 Add computed columns



Below is the result set after the two computed columns are added:

| | CLASS | STUDENTID | English | Math | PE | Total | Average |
|----|-----------|-----------|---------|------|----|-------|---------|
| 1 | Class one | 1 | 84 | 77 | 69 | 230 | 76.67 |
| 2 | Class one | 2 | 81 | 80 | 97 | 258 | 86.0 |
| 3 | Class one | 3 | 75 | 86 | 67 | 228 | 76.0 |
| 4 | Class one | 4 | 96 | 63 | 81 | 240 | 80.0 |
| 5 | Class one | 5 | 72 | 60 | 91 | 223 | 74.33 |
| 6 | Class one | 6 | 90 | 69 | 72 | 231 | 77.0 |
| 7 | Class one | 7 | 75 | 90 | 60 | 225 | 75.0 |
| 8 | Class one | 8 | 67 | 86 | 51 | 204 | 68.0 |
| 9 | Class one | 9 | 93 | 73 | 83 | 249 | 83.0 |
| 10 | Class one | 10 | 52 | 97 | 84 | 233 | 77.67 |
| 11 | Class one | 11 | 78 | 63 | 79 | 220 | 73.33 |
| 12 | Class one | 12 | 65 | 71 | 79 | 215 | 71.67 |
| 13 | Class one | 13 | 61 | 97 | 59 | 217 | 72.33 |
| 14 | Class one | 14 | 52 | 60 | 86 | 198 | 66.0 |
| 15 | Class two | 1 | 84 | 77 | 69 | 230 | 76.67 |

✦ 2.1 Add computed columns



Method 2: Add computed columns directly in the interface through writing expressions:

Open the scores table:

The screenshot shows a table with columns: CLASS, STUDENTID, English, Math, and PE. The PE column is highlighted in green. A toolbar with various icons is visible above the table. A yellow callout bubble points to the PE column header, and another yellow callout bubble points to the toolbar.

| | CLASS | STUDENTID | English | Math | PE |
|----|-----------|-----------|---------|------|----|
| 1 | Class one | 1 | 84 | 77 | 69 |
| 2 | Class one | 2 | 81 | 80 | 97 |
| 3 | Class one | 3 | 75 | 86 | 67 |
| 4 | Class one | 4 | 96 | 63 | 81 |
| 5 | Class one | 5 | 72 | 60 | 91 |
| 6 | Class one | 6 | 90 | 69 | 72 |
| 7 | Class one | 7 | 75 | 90 | 60 |
| 8 | Class one | 8 | 67 | 86 | 51 |
| 9 | Class one | 9 | 93 | 73 | 83 |
| 10 | Class one | 10 | 52 | 97 | 84 |
| 11 | Class one | 11 | 78 | 63 | 79 |
| 12 | Class one | 12 | 65 | 71 | 79 |
| 13 | Class one | 13 | 61 | 97 | 59 |
| 14 | Class one | 14 | 52 | 60 | 86 |
| 15 | Class two | 1 | 84 | 77 | 69 |
| 16 | Class two | 2 | 81 | 80 | 97 |

Click any column name, as the column operations toolbar will be activated only when a column is selected

Column operations toolbar

✦ 2.1 Add computed columns



Add a field through the toolbar icon:

| | CLASS | STUDENTID | English | Math | PE | Field |
|----|-----------|-----------|---------|------|----|--------|
| 1 | Class one | 1 | 84 | 77 | 69 | (null) |
| 2 | Class one | 2 | 81 | 80 | 97 | (null) |
| 3 | Class one | 3 | 75 | 86 | 67 | (null) |
| 4 | Class one | 4 | 96 | 63 | 81 | (null) |
| 5 | Class one | 5 | 72 | 60 | 91 | (null) |
| 6 | Class one | 6 | 90 | 69 | 72 | (null) |
| 7 | Class one | 7 | 75 | 90 | 60 | (null) |
| 8 | Class one | 8 | 67 | 86 | 51 | (null) |
| 9 | Class one | 9 | 93 | 73 | 83 | (null) |
| 10 | Class one | 10 | 52 | 97 | 84 | (null) |
| 11 | Class one | 11 | 78 | 63 | 79 | (null) |
| 12 | Class one | 12 | 65 | 71 | 79 | (null) |
| 13 | Class one | 13 | 61 | 97 | 59 | (null) |
| 14 | Class one | 14 | 52 | 60 | 86 | (null) |
| 15 | Class two | 1 | 84 | 77 | 69 | (null) |
| 16 | Class two | 2 | 81 | 80 | 97 | (null) |

✦ 2.1 Add computed columns



Enter a formula in the first row. Note that the double equal sign should be used to begin the formula:

| | CLASS | STUDENTID | English | Math | PE | Field |
|----|-----------|-----------|---------|------|----|--------------------------------|
| 1 | Class one | 1 | 84 | 77 | 69 | <code>==English+Math+PE</code> |
| 2 | Class one | 2 | 81 | 80 | 97 | (null) |
| 3 | Class one | 3 | 75 | 86 | 67 | (null) |
| 4 | Class one | 4 | 96 | 63 | 81 | (null) |
| 5 | Class one | 5 | 72 | 60 | 91 | (null) |
| 6 | Class one | 6 | 90 | 69 | 72 | (null) |
| 7 | Class one | 7 | 75 | 90 | 60 | (null) |
| 8 | Class one | 8 | 67 | 86 | 51 | (null) |
| 9 | Class one | 9 | 93 | 73 | 83 | (null) |
| 10 | Class one | 10 | 52 | 97 | 84 | (null) |
| 11 | Class one | 11 | 78 | 63 | 79 | (null) |
| 12 | Class one | 12 | 65 | 71 | 79 | (null) |
| 13 | Class one | 13 | 61 | 97 | 59 | (null) |
| 14 | Class one | 14 | 52 | 60 | 86 | (null) |
| 15 | Class two | 1 | 84 | 77 | 69 | (null) |

✦ 2.1 Add computed columns



Press carriage return to execute the formula and get the new computed column:

| | CLASS | STUDENTID | English | Math | PE | Field |
|----|-----------|-----------|---------|------|----|-------|
| 1 | Class one | 1 | 84 | 77 | 69 | 230 |
| 2 | Class one | 2 | 81 | 80 | 97 | 258 |
| 3 | Class one | 3 | 75 | 86 | 67 | 228 |
| 4 | Class one | 4 | 96 | 63 | 81 | 240 |
| 5 | Class one | 5 | 72 | 60 | 91 | 223 |
| 6 | Class one | 6 | 90 | 69 | 72 | 231 |
| 7 | Class one | 7 | 75 | 90 | 60 | 225 |
| 8 | Class one | 8 | 67 | 86 | 51 | 204 |
| 9 | Class one | 9 | 93 | 73 | 83 | 249 |
| 10 | Class one | 10 | 52 | 97 | 84 | 233 |
| 11 | Class one | 11 | 78 | 63 | 79 | 220 |
| 12 | Class one | 12 | 65 | 71 | 79 | 215 |
| 13 | Class one | 13 | 61 | 97 | 59 | 217 |
| 14 | Class one | 14 | 52 | 60 | 86 | 198 |
| 15 | Class two | 1 | 84 | 77 | 69 | 230 |

✦ 2.1 Add computed columns



Rename the computed column through the pop-up menu:

| | CLASS | STUDENTID | English | Math | PE | Field |
|----|-----------|-----------|---------|------|----|-------|
| 1 | Class one | 1 | 84 | 77 | 69 | |
| 2 | Class one | 2 | 81 | 80 | 97 | |
| 3 | Class one | 3 | 75 | 86 | 67 | |
| 4 | Class one | 4 | 96 | 63 | 81 | |
| 5 | Class one | 5 | 72 | 60 | 91 | |
| 6 | Class one | 6 | 90 | 69 | 72 | |
| 7 | Class one | 7 | 75 | 90 | 60 | |
| 8 | Class one | 8 | 67 | 86 | 51 | |
| 9 | Class one | 9 | 93 | 73 | 83 | 249 |
| 10 | Class one | 10 | 52 | 97 | 84 | 233 |
| 11 | Class one | 11 | 78 | 63 | 79 | 220 |
| 12 | Class one | 12 | 65 | 71 | 79 | 215 |
| 13 | Class one | 13 | 61 | 97 | 59 | 217 |
| 14 | Class one | 14 | 52 | 60 | 86 | 198 |
| 15 | Class two | 1 | 84 | 77 | 69 | 230 |

Click column name to select the column and execute **"Rename field"** on the pop-up menu

✦ 2.1 Add computed columns



Enter a new field name to finish adding the computed column:

Input

Please enter a new name

OK Cancel

| | CLASS | STUDENTID | English | Math | PE | Total |
|----|-----------|-----------|---------|------|----|-------|
| 1 | Class one | 1 | 84 | 77 | 69 | 230 |
| 2 | Class one | 2 | 81 | 80 | 97 | 258 |
| 3 | Class one | 3 | 75 | 86 | 67 | 228 |
| 4 | Class one | 4 | 96 | 63 | 81 | 240 |
| 5 | Class one | 5 | 72 | 60 | 91 | 223 |
| 6 | Class one | 6 | 90 | 69 | 72 | 231 |
| 7 | Class one | 7 | 75 | 90 | 60 | 225 |
| 8 | Class one | 8 | 67 | 86 | 51 | 204 |
| 9 | Class one | 9 | 93 | 73 | 83 | 249 |
| 10 | Class one | 10 | 52 | 97 | 84 | 233 |
| 11 | Class one | 11 | 78 | 63 | 79 | 220 |
| 12 | Class one | 12 | 65 | 71 | 79 | 215 |
| 13 | Class one | 13 | 61 | 97 | 59 | 217 |
| 14 | Class one | 14 | 52 | 60 | 86 | 198 |
| 15 | Class two | 1 | 84 | 77 | 69 | 230 |

CONTENTS



Filter data

✦ 2.2 Filter data



Below is Excel table `stock.2.2.xlsx` recording stock opening indexes in May – June, 2020. We need to find the dates when the opening prices are greater than 2900:

| | A | B | C |
|----|------------|---------|---|
| 1 | date | open | |
| 2 | 2020/06/22 | 2966.9 | |
| 3 | 2020/06/19 | 2938.79 | |
| 4 | 2020/06/18 | 2929.88 | |
| 5 | 2020/06/17 | 2932.67 | |
| 6 | 2020/06/16 | 2912.83 | |
| 7 | 2020/06/15 | 2908.28 | |
| 8 | 2020/06/12 | 2876.8 | |
| 9 | 2020/06/11 | 2939.79 | |
| 10 | 2020/06/10 | 2951.28 | |
| 11 | 2020/06/09 | 2939.54 | |
| 12 | 2020/06/08 | 2941.98 | |
| 13 | 2020/06/05 | 2923.19 | |
| 14 | 2020/06/04 | 2931.84 | |
| 15 | 2020/06/03 | 2930.39 | |
| 16 | 2020/06/02 | 2916.32 | |
| 17 | 2020/06/01 | 2871.96 | |
| 18 | 2020/05/29 | 2835.58 | |
| 19 | 2020/05/28 | 2838.21 | |

✦ 2.2 Filter data



Double-click `stock.2.2.xlsx` in `data/2` to open the data table named `stock`:

A screenshot of a data table interface. At the top, there's a 'Result' label and a 'stock' tab. Below the tab is a toolbar with several icons: a filter icon (a blue funnel) which is circled in red, a sort icon (A-Z with a downward arrow), a refresh icon, a chart icon, and a copy icon. Below the toolbar is a table with columns 'date' and 'open'. The table contains 11 rows of data. A yellow callout bubble points from the filter icon to the text 'Then click Filter icon'.

Then click **Filter**
icon

✦ 2.2 Filter data



Set up the following filter expression on “Filter” edit interface:

The screenshot shows a 'Filter' dialog box with the following fields and controls:

- Name: filter
- Source table: stock
- Filter expression: 1 open > 2900 (highlighted with a red box)
- Field (Double click to select): date, open (selected)
- Value (Double click to select): 2816.24, 2827.9, 2831.63, 2835.58, 2838.21, 2847.32, 2863.05, 2871.96, 2872.52, 2876.47, 2876.8, 2880.71, 2882.71, 2882.96
- Operator: +, -, *, /, <, >, <=, >=, =, !=, (,), AND, OR, NOT, ==
- Buttons: OK, Cancel

✦ 2.2 Filter data



Execute the filtering operation to get the following result:

The screenshot shows a software interface for a data table. At the top, there is a 'Result' label, a minus sign icon, and a pencil icon. Below this are two tabs: 'stock' and 'filter'. Under the 'filter' tab, there is a row of icons: 'ABC 123', a funnel (filter), 'A Z' with a downward arrow, a tree structure, a grid, and a document icon. The table below has three columns: 'date' and 'open' are highlighted in grey. The 'date' column contains dates from 2020-06-22 down to 2020-06-05. The 'open' column contains numerical values from 2966.9 down to 2923.19. There are 11 rows in total, numbered 1 to 11 in the first column.

| | date | open |
|----|------------|---------|
| 1 | 2020-06-22 | 2966.9 |
| 2 | 2020-06-19 | 2938.79 |
| 3 | 2020-06-18 | 2929.88 |
| 4 | 2020-06-17 | 2932.67 |
| 5 | 2020-06-16 | 2912.83 |
| 6 | 2020-06-15 | 2908.28 |
| 7 | 2020-06-11 | 2939.79 |
| 8 | 2020-06-10 | 2951.28 |
| 9 | 2020-06-09 | 2939.54 |
| 10 | 2020-06-08 | 2941.98 |
| 11 | 2020-06-05 | 2923.19 |

CONTENTS



Sort data

✦ 2.3 Sort data



In section 2.1, a total score field named Total was added to the student score table.

For the convenience of checking the scores, we want to sort rows by total score (Total) field in descending order:

| | CLASS | STUDENTID | English | Math | PE | Total |
|----|-----------|-----------|---------|------|----|-------|
| 1 | Class one | 1 | 84 | 77 | 69 | 230 |
| 2 | Class one | 2 | 81 | 80 | 97 | 258 |
| 3 | Class one | 3 | 75 | 86 | 67 | 228 |
| 4 | Class one | 4 | 96 | 63 | 81 | 240 |
| 5 | Class one | 5 | 72 | 60 | 91 | 223 |
| 6 | Class one | 6 | 90 | 69 | 72 | 231 |
| 7 | Class one | 7 | 75 | 90 | 60 | 225 |
| 8 | Class one | 8 | 67 | 86 | 51 | 204 |
| 9 | Class one | 9 | 93 | 73 | 83 | 249 |
| 10 | Class one | 10 | 52 | 97 | 84 | 233 |
| 11 | Class one | 11 | 78 | 63 | 79 | 220 |

Click **Sort** icon above the result table containing the Total field

✦ 2.3 Sort data



In the “Sort” edit interface, add a sorting field in the place highlighted by the red square:

The screenshot shows the 'Sort' dialog box with the following details:

- Name: scores
- Source table: scores
- Locale: English
- Options: (empty)
- Sorting field: (empty)
- Source fields (Double click to select): CLASS, English, Math, PE, STUDENTID, Total
- Parallel computation:
- Place null values at the end:

| Index | Field | Ascending |
|-------|-------|--------------------------|
| 1 | Total | <input type="checkbox"/> |

Note: If the source table is a cursor, the sorting field can only be arranged in ascending order and sorting direction cannot be edited

✦ 2.3 Sort data



Execute the sorting operation and get the following result:

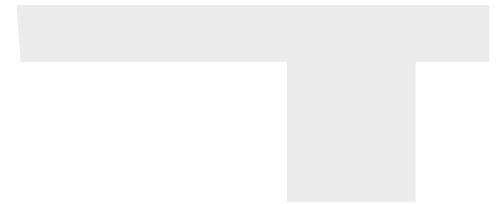
| | CLASS | STUDENTID | English | Math | PE | Total |
|----|-----------|-----------|---------|------|----|-------|
| 1 | Class one | 2 | 81 | 80 | 97 | 258 |
| 2 | Class two | 2 | 81 | 80 | 97 | 258 |
| 3 | Class one | 9 | 93 | 73 | 83 | 249 |
| 4 | Class two | 9 | 93 | 73 | 83 | 249 |
| 5 | Class one | 4 | 96 | 71 | 81 | 240 |
| 6 | Class two | 4 | 96 | 63 | 81 | 240 |
| 7 | Class one | 10 | 52 | 97 | 84 | 233 |
| 8 | Class two | 10 | 52 | 97 | 84 | 233 |
| 9 | Class one | 6 | 90 | 69 | 72 | 231 |
| 10 | Class two | 6 | 90 | 69 | 72 | 231 |
| 11 | Class one | 1 | 84 | 77 | 69 | 230 |

Sort rows by total score in descending order to get the top 3 in grade

CONTENTS



Group data



✦ 2.4 Group data



Below is sale orders table `orders.2.4.btx`. We want to compute order amount for each seller.

| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 1 | UJRNP | 17 | 392.0 | 2012-11-02 15:28:05 |
| | 2 | SJCH | 6 | 4802.0 | 2012-11-09 15:28:05 |
| 3 | 3 | UJRNP | 16 | 13500.0 | 2012-11-05 15:28:05 |
| 4 | 4 | PWQ | 9 | 26100.0 | 2012-11-08 15:28:05 |
| 5 | 5 | PWQ | 11 | 4410.0 | 2012-11-12 15:28:05 |
| 6 | 6 | HANAR | 18 | 6174.0 | 2012-11-07 15:28:05 |
| 7 | 7 | EGU | 2 | 17800.0 | 2012-11-06 15:28:05 |
| 8 | 8 | VILJX | 7 | 2156.0 | 2012-11-09 15:28:05 |
| 9 | 9 | JAYB | 14 | 17400.0 | 2012-11-12 15:28:05 |
| 10 | 10 | JAXE | 19 | 19200.0 | 2012-11-12 15:28:05 |
| 11 | 11 | SJCH | 7 | 13700.0 | 2012-11-10 15:28:05 |

Click **Group** icon

✦ 2.4 Group data



On the “Group” edit interface, set up grouping field and aggregate expression, as shown by the red squares:

The image shows two overlapping screenshots of the "Group" dialog box in a data tool. The left screenshot shows the "Grouping field" tab, where the "Field expression" table has one row with "SELLERID" highlighted by a red square. The right screenshot shows the "Aggregates" tab, where the "Aggregate expression" table has one row with "sum(AMOUNT)" and "GrandTotal" highlighted by a red square. The "Aggregates" tab label is also circled in red.

Left Screenshot (Grouping field tab):

| Index | Field expression |
|-------|------------------|
| 1 | SELLERID |

Right Screenshot (Aggregates tab):

| Index | Aggregate expression | Alias |
|-------|----------------------|------------|
| 1 | sum(AMOUNT) | GrandTotal |

✦ 2.4 Group data



Execute the grouping operation and get the following result:

| | SELLERID | GrandTotal |
|----|----------|------------|
| 1 | 1 | 687850.0 |
| 2 | 2 | 734276.0 |
| 3 | 3 | 554724.0 |
| 4 | 4 | 565594.0 |
| 5 | 5 | 564516.0 |
| 6 | 6 | 672064.0 |
| 7 | 7 | 504764.0 |
| 8 | 8 | 602648.0 |
| 9 | 9 | 596578.0 |
| 10 | 10 | 541826.0 |
| 11 | 11 | 687894.0 |

✦ 2.4 Group data



Different from SQL that must specify the aggregate expression for each grouping operation, SPL supports returning a set after data is grouped. As the following shows, set up the grouping field and select “Retain details” option:

The screenshot shows the 'Group' dialog box with the following configuration:

- Name: group1
- Source table: orders
- Options: (empty)
- Aggregates: sum
- Grouping field: SELLERID
- Source fields: AMOUNT, CLIENT, ORDERDATE, ORDERID, SELLERID (selected)
- Retain details:

| Index | Field expression |
|-------|------------------|
| 1 | SELLERID |

◆ 2.4 Group data



Execute the grouping operation and a Details field is generated. In the field, each group of rows is a set:

| | SELLERID | Details |
|----|----------|---------------------------------|
| 1 | 1 | [[14,JAYB,1, ...],[77,HANAR,1, |
| 2 | 2 | [[7,EGU,2, ...],[19,JOPO,2, |
| 3 | 3 | [[17,PJIPE,3, ...],[22,JAXE,3, |
| 4 | 4 | 16,AYWYN,4, |
| 5 | 5 | [[21,DILRT,5, ...],[34,HANAR,5, |
| 6 | 6 | [[2,SJCH,6, ...],[31,QHHW,6, |
| 7 | 7 | [[8,VILJX,7, ...],[11,SJCH,7, |
| 8 | 8 | [[20,EGU,8, ...],[24,FHYBR,8, |
| 9 | 9 | [[4,PWQ,9, ...],[32,SAVEA,9, |
| 10 | 10 | [[29,QHHW,10, |
| 11 | 11 | [[5,PWQ,11, ...],[12,QUICK,11, |

Double-click
Details field to
open a
subtable and
view its data

| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 16 | AYWYN | 4 | 6566.0 | 2012-11-21 15:28:05 |
| 2 | 37 | ERNSH | 4 | 98.0 | 2012-12-07 15:28:05 |
| 3 | 90 | EGU | 4 | 26700.0 | 2013-01-30 15:28:05 |
| 4 | 118 | AVU | 4 | 1764.0 | 2013-02-26 15:28:05 |
| 5 | 134 | JOPO | 4 | 7050.0 | 2013-03-19 15:28:05 |
| 6 | 165 | HANAR | 4 | 1528.05 | 2013-03-19 15:28:05 |
| 7 | | | | | |
| 8 | | | | | |
| 9 | | | | | |
| 10 | | | | | |
| 11 | | | | | |

A table in the form of P.F(r) is details table, where P is the parent table, F is a field name of the parent table, and r is the record number. Separators should be removed when the table is referenced at the command line; it is written as PFr, such as `group1Details4`

CONTENTS



Concatenate detailed data

✦ 2.5 Concatenate detailed data



The detailed data field Details generated from the grouping operation can be concatenated as a table sequence:

| | SELLERID | Details |
|----|----------|---------------------------------|
| 1 | | [[14,JAYB,1, ...],[77,HANAR,1, |
| 2 | 2 | [[7,EGU,2, ...],[19,JOPO,2, |
| 3 | 3 | [[17,PJPE,3, ...],[22,JAXE,3, |
| 4 | 4 | [[16,AYWYN,4, |
| 5 | 5 | [[21,DILRT,5, ...],[34,HANAR,5, |
| 6 | 6 | [[2,SJCH,6, ...],[31,QHHW,6, |
| 7 | 7 | [[8,VILJX,7, ...],[11,SJCH,7, |
| 8 | 8 | [[20,EGU,8, ...],[24,FHYBR,8, |
| 9 | 9 | [[4,PWQ,9, ...],[32,SAVEA,9, |
| 10 | 10 | [[29,QHHW,10, |
| 11 | 11 | [[5,PWQ,11, ...],[12,QUICK,11, |
| 12 | 12 | [[13,HL,12, ...],[38,DILRT,12, |
| 13 | 13 | [[54,JAXE,13, ...],[64,HP,13, |
| 14 | 14 | [[9,JAYB,14, ...],[39,GLH,14, |
| 15 | 15 | [[28,AVU,15, ...],[114,PWQ,15, |

Click
Concatenate
detailed data icon

✦ 2.5 Concatenate detailed data



On the “Concatenate detailed data” edit interface, select Details field highlighted in red square:

A screenshot of a software dialog box titled "Concatenate grouped result". The dialog has a close button (X) in the top right corner. It contains several input fields: "Name" with the value "conj", "Source table" with the value "group", and "Detail field" with a dropdown menu showing "Details". The "Detail field" dropdown is highlighted with a red square. Below these fields is a checkbox labeled "Recursive computation" which is currently unchecked. On the right side of the dialog, there are "OK" and "Cancel" buttons.

Concatenate grouped result

Name conj Source table group OK

Detail field Details Options Cancel

Recursive computation

You can only select the Details field from the drop-down field list

✦ 2.5 Concatenate detailed data



Concatenate Details field as table sequence conj. Below is the result data:

| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 14 | JAYB | 1 | 7644.0 | 2012-11-16 15:28:05 |
| 2 | 77 | HANAR | 1 | 13200.0 | 2013-01-17 15:28:05 |
| 3 | 78 | YZ | 1 | 11600.0 | 2013-01-20 15:28:05 |
| 4 | 93 | AVU | 1 | 21800.0 | 2013-02-05 15:28:05 |
| 5 | 104 | HL | 1 | 26400.0 | 2013-02-18 15:28:05 |
| 6 | 109 | PWQ | 1 | 17500.0 | 2013-02-21 15:28:05 |
| 7 | 120 | FHYBR | 1 | 16000.0 | 2013-03-03 15:28:05 |
| 8 | 127 | HP | 1 | 13600.0 | 2013-03-15 15:28:05 |
| 9 | 189 | DNEDL | 1 | 26100.0 | 2013-05-13 15:28:05 |
| 10 | 200 | EGU | 1 | 14000.0 | 2013-05-20 15:28:05 |
| 11 | 201 | DNEDL | 1 | 7350.0 | 2013-05-25 15:28:05 |

CONTENTS



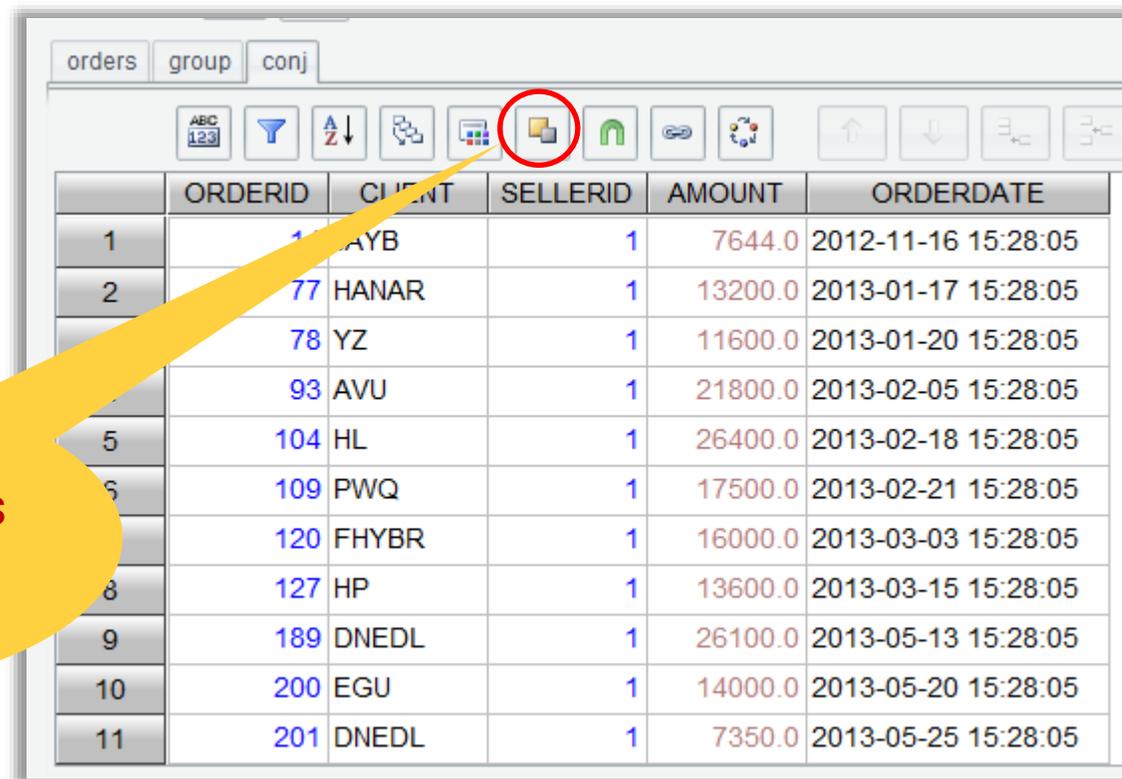
2.6

Select fields

✦ 2.6 Select fields



Select desired fields through “Select fields” option. For example, to select SELLERID and AMOUNT from data table conj:



| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 1 | AYB | 1 | 7644.0 | 2012-11-16 15:28:05 |
| 2 | 77 | HANAR | 1 | 13200.0 | 2013-01-17 15:28:05 |
| | 78 | YZ | 1 | 11600.0 | 2013-01-20 15:28:05 |
| | 93 | AVU | 1 | 21800.0 | 2013-02-05 15:28:05 |
| 5 | 104 | HL | 1 | 26400.0 | 2013-02-18 15:28:05 |
| 6 | 109 | PWQ | 1 | 17500.0 | 2013-02-21 15:28:05 |
| | 120 | FHYBR | 1 | 16000.0 | 2013-03-03 15:28:05 |
| 8 | 127 | HP | 1 | 13600.0 | 2013-03-15 15:28:05 |
| 9 | 189 | DNEDL | 1 | 26100.0 | 2013-05-13 15:28:05 |
| 10 | 200 | EGU | 1 | 14000.0 | 2013-05-20 15:28:05 |
| 11 | 201 | DNEDL | 1 | 7350.0 | 2013-05-25 15:28:05 |

Click **Select fields**
icon

✦ 2.6 Select fields



As the following shows, add expressions of fields to be selected:

| Index | Expression | Alias |
|-------|------------|----------|
| 1 | SELLERID | SELLERID |
| 2 | AMOUNT | AMOUNT |

Similar to adding computed columns, a to-be-selected field can be specified by an expression. Their difference is that the former is to add fields to the source table and "Select fields" is to generate a new table sequence through setting up expressions

✦ 2.6 Select fields



Execute field selection operation and get the following result:

| | SELLERID | AMOUNT |
|----|----------|---------|
| 1 | 1 | 7644.0 |
| 2 | 1 | 13200.0 |
| 3 | 1 | 11600.0 |
| 4 | 1 | 21800.0 |
| 5 | 1 | 26400.0 |
| 6 | 1 | 17500.0 |
| 7 | 1 | 16000.0 |
| 8 | 1 | 13600.0 |
| 9 | 1 | 26100.0 |
| 10 | 1 | 14000.0 |
| 11 | 1 | 7350.0 |

CONTENTS

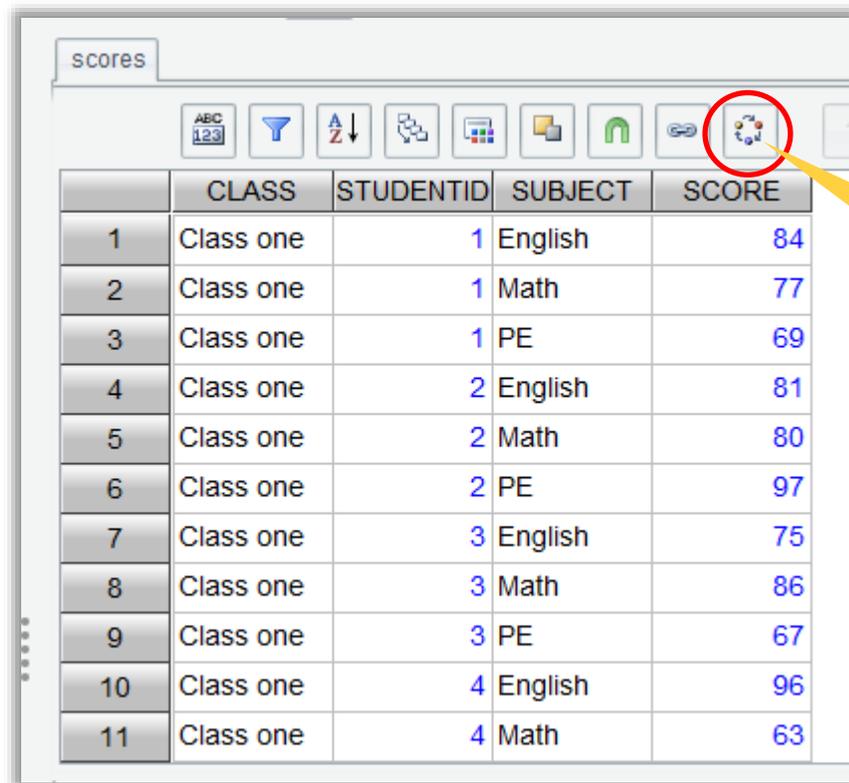


Transpose rows/ columns
to /columns/rows

✦ 2.7 Transpose rows/columns to columns/rows



Generally, courses for students from different majors are dynamic and not suitable to be used as field names, but a report using courses as column headers is preferred. Below is opened student score table `scores.2.7.txt` :



| | 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 |
| 11 | Class one | 4 | Math | 63 |

Click **Transpose row to column** icon

Note: A cursor type source table does not support this operation

✦ 2.7 Transpose rows/columns to columns/rows



Set up the following values on “Pivot” interface:

The screenshot shows the 'Pivot' dialog box with the following configuration:

- Name: pivot
- Source table: scores
- Field column: SUBJECT
- Data column: SCORE
- Options: (empty)
- Transposed column to row:
- Grouping field: (empty)
- Source fields (Double click to select): CLASS, SCORE, STUDENTID, SUBJECT

| Index | Field expression |
|-------|------------------|
| 1 | CLASS |
| 2 | STUDENTID |

✦ 2.7 Transpose rows/columns to columns/rows



Transpose rows to columns and get the following result table:

| | CLASS | STUDENTID | English | Math | PE |
|----|-----------|-----------|---------|------|----|
| 1 | Class one | 1 | 84 | 77 | 69 |
| 2 | Class one | 2 | 81 | 80 | 97 |
| 3 | Class one | 3 | 75 | 86 | 67 |
| 4 | Class one | 4 | 96 | 63 | 81 |
| 5 | Class one | 5 | 72 | 60 | 91 |
| 6 | Class one | 6 | 90 | 69 | 72 |
| 7 | Class one | 7 | 75 | 90 | 60 |
| 8 | Class one | 8 | 67 | 86 | 51 |
| 9 | Class one | 9 | 93 | 73 | 83 |
| 10 | Class one | 10 | 52 | 97 | 84 |
| 11 | Class one | 11 | 78 | 63 | 79 |

✦ 2.7 Transpose rows/columns to columns/rows



We can also rotate the pivot table by setting up these values as follows:

Name: pivotReverse

Field column: SUBJECT

Data column: SCORE

Transpose column to row

Options: r

Source table: pivot

Source fields (Double click to select): CLASS, English, Math, PE, STUDENTID

| Index | Field expression |
|-------|------------------|
| 1 | CLASS |
| 2 | STUDENTID |

Select "Transpose column to row" checkbox

Note: For column-to-row transposition, drop-down lists of "Field column" and "Data column" cannot be displayed and the desired values need to be manually typed in because the two names are to-be-generated columns

✦ 2.7 Transpose rows/columns to columns/rows



Transpose columns to rows and get the following result set:

| | 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 |
| 11 | Class one | 4 | Math | 63 |

CONTENTS



Data association

✦ 2.8 Data association



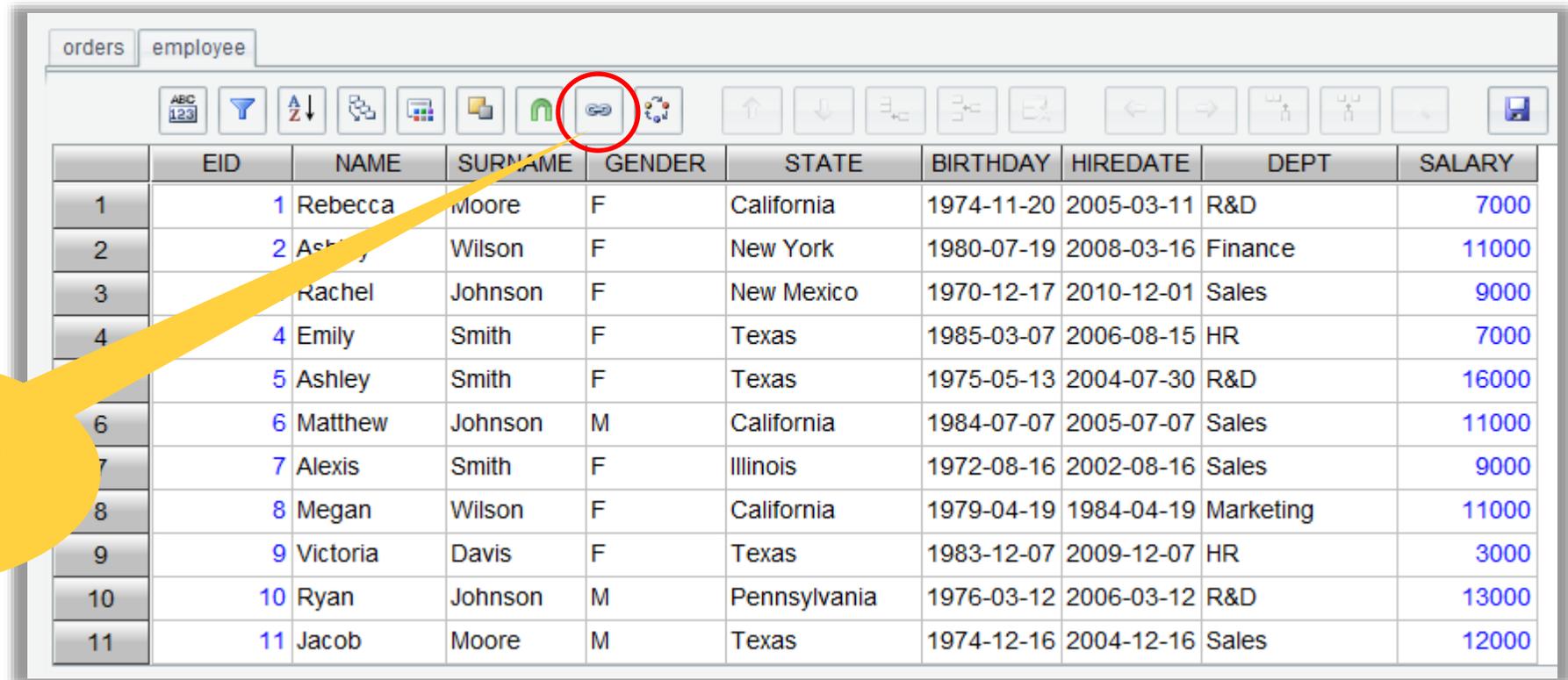
In the orders table in section 2.4, sellers are stored as IDs, which are inconvenient to view. Open data table [orders.2.4.btx](#) and its data is as follows:

| | ORDERID | CLIENT | SELLERID | AMOUNT | ORDERDATE |
|----|---------|--------|----------|---------|---------------------|
| 1 | 1 | UJRN | 17 | 392.0 | 2012-11-02 15:28:05 |
| 2 | 2 | SJCH | 6 | 4802.0 | 2012-11-09 15:28:05 |
| 3 | 3 | UJRN | 16 | 13500.0 | 2012-11-05 15:28:05 |
| 4 | 4 | PWQ | 9 | 26100.0 | 2012-11-08 15:28:05 |
| 5 | 5 | PWQ | 11 | 4410.0 | 2012-11-12 15:28:05 |
| 6 | 6 | HANAR | 18 | 6174.0 | 2012-11-07 15:28:05 |
| 7 | 7 | EGU | 2 | 17800.0 | 2012-11-06 15:28:05 |
| 8 | 8 | VILJX | 7 | 2156.0 | 2012-11-09 15:28:05 |
| 9 | 9 | JAYB | 14 | 17400.0 | 2012-11-12 15:28:05 |
| 10 | 10 | JAXE | 19 | 19200.0 | 2012-11-12 15:28:05 |
| 11 | 11 | SJCH | 7 | 13700.0 | 2012-11-10 15:28:05 |

✦ 2.8 Data association



Then open employees table **employee.2.8.xlsx**, whose data is as follows:



The screenshot shows a data table with 11 rows and 10 columns. The columns are labeled EID, NAME, SURNAME, GENDER, STATE, BIRTHDAY, HIREDATE, DEPT, and SALARY. The 'Join' icon in the toolbar is circled in red. A yellow callout bubble points to this icon with the text 'Click Join icon'.

| | 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 |
| 9 | 9 | Victoria | Davis | F | Texas | 1983-12-07 | 2009-12-07 | HR | 3000 |
| 10 | 10 | Ryan | Johnson | M | Pennsylvania | 1976-03-12 | 2006-03-12 | R&D | 13000 |
| 11 | 11 | Jacob | Moore | M | Texas | 1974-12-16 | 2004-12-16 | Sales | 12000 |

Click **Join**
icon

✦ 2.8 Data association



Select “orders” table under “Target table” panel:

The screenshot shows a 'Join' dialog box with the following fields and options:

- Name: join1
- Source table: employee
- Join type: Inner join, Left join
- Target table: orders (highlighted in yellow)
- Join field: (empty)
- Target fields: (empty)
- Select: (circled in red)

| Index | Target table | Select |
|-------|--------------|-------------------------------------|
| 1 | orders | <input checked="" type="checkbox"/> |

✦ 2.8 Data association



And then set up join fields and select desired fields:

Join

Name: Source table:

Join type:
 Inner join Left join

Target table | Join field | Target fields

| Index | employee | orders |
|-------|----------|----------|
| 1 | EID | SELLERID |

Join

Name: Source table:

Join type:
 Inner join Left join

Target table | Join field | Target fields

| Index | Table | Field | Select | Alias |
|-------|----------|-----------|-------------------------------------|-------|
| 1 | employee | EID | <input type="checkbox"/> | |
| 2 | employee | NAME | <input checked="" type="checkbox"/> | |
| 3 | employee | SURNAME | <input type="checkbox"/> | |
| 4 | employee | GENDER | <input type="checkbox"/> | |
| 5 | employee | STATE | <input type="checkbox"/> | |
| 6 | employee | BIRTHDAY | <input type="checkbox"/> | |
| 7 | employee | HIREDATE | <input type="checkbox"/> | |
| 8 | employee | DEPT | <input type="checkbox"/> | |
| 9 | employee | SALARY | <input type="checkbox"/> | |
| 10 | orders | ORDERID | <input checked="" type="checkbox"/> | |
| 11 | orders | CLIENT | <input checked="" type="checkbox"/> | |
| 12 | orders | SELLERID | <input type="checkbox"/> | |
| 13 | orders | AMOUNT | <input checked="" type="checkbox"/> | |
| 14 | orders | ORDERDATE | <input checked="" type="checkbox"/> | |

✦ 2.8 Data association



Perform data association and get the following result set:

| | NAME | ORDERID | CLIENT | AMOUNT | ORDERDATE |
|----|---------|---------|--------|---------|---------------------|
| 1 | Rebecca | 14 | JAYB | 7644.0 | 2012-11-16 15:28:05 |
| 2 | Rebecca | 78 | YZ | 11600.0 | 2013-01-20 15:28:05 |
| 3 | Rebecca | 104 | HL | 26400.0 | 2013-02-18 15:28:05 |
| 4 | Rebecca | 120 | FHYBR | 16000.0 | 2013-03-03 15:28:05 |
| 5 | Rebecca | 189 | DNEDL | 26100.0 | 2013-05-13 15:28:05 |
| 6 | Rebecca | 201 | DNEDL | 7350.0 | 2013-05-25 15:28:05 |
| 7 | Rebecca | 237 | PWQ | 23400.0 | 2013-06-24 15:28:05 |
| 8 | Rebecca | 278 | FHYBR | 23900.0 | 2013-08-12 15:28:05 |
| 9 | Rebecca | 288 | UJRNP | 4998.0 | 2013-08-22 15:28:05 |
| 10 | Rebecca | 380 | JAYB | 13000.0 | 2013-11-17 15:28:05 |
| 11 | Rebecca | 401 | DILRT | 20200.0 | 2013-12-08 15:28:05 |

It is
sellers' names
that are
displayed

CONTENTS



✦ 2.9 Set operations



`cities.2.9.xlsx` stores two sheets of city population data; they have some common records. We need to organize them to get all unique city population records. Here are the two sheets:

| | A | B | C | D |
|----|-----|------------------|------------|---------|
| 1 | CID | NAME | POPULATION | STATEID |
| 2 | 101 | Durham | 209009 | 33 |
| 3 | 75 | Buffalo | 276059 | 32 |
| 4 | 54 | El Paso | 609415 | 43 |
| 5 | 97 | Chula Vista | 212756 | 5 |
| 6 | 5 | Philadelphia | 1492231 | 38 |
| 7 | 62 | Raleigh | 356321 | 33 |
| 8 | 32 | Albuquerque | 463874 | 31 |
| 9 | 30 | Cleveland | 467851 | 35 |
| 10 | 103 | Modesto | 205721 | 5 |
| 11 | 70 | Riverside | 293761 | 5 |
| 12 | 44 | Colorado Springs | 371182 | 6 |
| 13 | 4 | Houston | 2009834 | 43 |
| 14 | 40 | Tulsa | 391908 | 36 |
| 15 | 79 | Fort Wayne | 248637 | 14 |
| 16 | 45 | St. Louis | 338353 | 25 |
| 17 | 22 | Denver | 560415 | 6 |
| 18 | 66 | Cincinnati | 332252 | 35 |
| 19 | 86 | Norfolk | 238832 | 46 |

| | A | B | C | D |
|----|-----|------------------|------------|---------|
| 1 | CID | NAME | POPULATION | STATEID |
| 2 | 75 | Buffalo | 276059 | 32 |
| 3 | 50 | Cincinnati | 323885 | 35 |
| 4 | 3 | Chicago | 2886251 | 13 |
| 5 | 18 | Boston | 589281 | 21 |
| 6 | 85 | Chandler | 240595 | 3 |
| 7 | 109 | Boise | 198638 | 12 |
| 8 | 88 | Scottsdale | 231127 | 3 |
| 9 | 6 | Phoenix | 1371960 | 3 |
| 10 | 11 | San Jose | 900443 | 5 |
| 11 | 27 | Tucson | 503151 | 3 |
| 12 | 71 | Stockton | 290141 | 5 |
| 13 | 44 | Colorado Springs | 371182 | 6 |
| 14 | 99 | Reno | 210255 | 28 |
| 15 | 66 | Cincinnati | 332252 | 35 |
| 16 | 81 | Glendale | 246531 | 3 |
| 17 | 73 | Newark | 281402 | 30 |
| 18 | 22 | Denver | 560415 | 6 |
| 19 | 79 | Fort Wayne | 248637 | 14 |

✦ 2.9 Set operations



Open the two sheets through “Browse” button:

| | CID | NAME | POPULATION | STATEID |
|----|-----|------------|------------|---------|
| 1 | 75 | Buffalo | 276059 | 32 |
| 2 | 50 | Cincinnati | 323885 | 35 |
| 3 | 3 | Chicago | 2886251 | 13 |
| 4 | 18 | Boston | 589281 | 21 |
| 5 | 85 | Chandler | 240595 | 3 |
| 6 | 109 | Boise | 198638 | 12 |
| 7 | 88 | Scottsdale | 231127 | 3 |
| 8 | 6 | Phoenix | 1371960 | 3 |
| 9 | 11 | San Jose | 900443 | 5 |
| 10 | 27 | Tucson | 503151 | 3 |
| 11 | 71 | Stockton | 290141 | 5 |

Click **set operations** icon

✦ 2.9 Set operations



Do the following configurations in “Set operations” edit interface:

Set operations

Name: setoperator Source table: Sheet2

Operator:

Intersection Union Union all Difference

Target table: Target field:

| Index | Table | Select |
|-------|--------|-------------------------------------|
| 1 | Sheet1 | <input checked="" type="checkbox"/> |

Only tables having same structure as the source table are listed.

Select **Union** operation

✦ 2.9 Set operations



Perform union operation and get the following result set:

| | CID | NAME | POPULATION | STATEID |
|----|-----|------------|------------|---------|
| 1 | 75 | Buffalo | 276059 | 32 |
| 2 | 50 | Cincinnati | 323885 | 35 |
| 3 | 3 | Chicago | 2886251 | 13 |
| 4 | 18 | Boston | 589281 | 21 |
| 5 | 85 | Chandler | 240595 | 3 |
| 6 | 109 | Boise | 198638 | 12 |
| 7 | 88 | Scottsdale | 231127 | 3 |
| 8 | 6 | Phoenix | 1371960 | 3 |
| 9 | 11 | San Jose | 900443 | 5 |
| 10 | 27 | Tucson | 503151 | 3 |
| 11 | 71 | Stockton | 290141 | 5 |

SPL WIN Course

Chapter 3

Comprehensive scenarios

SPL WIN



CONTENTS



3.1 Aggregation on a computed column in a grouped table

3.2 Intragroup sorting on a grouped table

3.3 Intragroup filtering on a grouped table

3.4 Retain the first row of each group

CONTENTS



3.5

Grouping & aggregation after filtering

3.6

Compute link relative in a multilayer-structure table

3.7

Comparison between multiple tables

CONTENTS



Aggregation on a computed column in a grouped table

◆ 3.1 Aggregation on a computed column in a grouped table



Below is a food sales table ordered by food types. We need to compute the sales amount of each food item, the total sales amount of each type of food, and that of all foods. Below is the table (**food.3.1.xlsx**):

| | A | B | C | D | E |
|----|----------------|-----------|-----------|------|----------|
| 1 | Commodity | Type | UnitPrice | Unit | Quantity |
| 2 | Milk | Drink | 3.99 | GAL | 50 |
| 3 | Coke | Drink | 0.9 | L | 300 |
| 4 | Juice | Drink | 0.5 | L | 120 |
| 5 | Apple | Fruit | 0.99 | LB | 150.5 |
| 6 | Avocado | Fruit | 1.49 | EA | 77 |
| 7 | Pineapple | Fruit | 1.49 | LB | 30.99 |
| 8 | Peach | Fruit | 0.88 | LB | 118.99 |
| 9 | Orange | Fruit | 0.49 | LB | 277.25 |
| 10 | Pork Spareribs | Meat | 1.99 | LB | 58.45 |
| 11 | Drum stick | Meat | 0.99 | LB | 35.88 |
| 12 | Fresh Salmon | Seafood | 7.99 | LB | 25.66 |
| 13 | Razor Clam | Seafood | 3.99 | LB | 33.89 |
| 14 | Shrimp | Seafood | 15.99 | LB | 20.75 |
| 15 | Scallop | Seafood | 4.99 | LB | 66.5 |
| 16 | Spinach | Vegetable | 0.89 | LB | 123.35 |
| 17 | Lettuce | Vegetable | 0.79 | LB | 133 |
| 18 | | | | | |

3.1 Aggregation on a computed column in a grouped table



Open the data table, add a computed column, and compute sales amount of each food:

Here is a long expression, you can directly copy it during learning

Expression: UnitPrice*Quantity

| Index | Expression | Alias |
|-------|--------------------|--------|
| 1 | UnitPrice*Quantity | Amount |

| Commodity | Type | UnitPrice | Unit |
|------------------|---------|-----------|------|
| 1 Milk | Drink | 3.99 | GAL |
| 2 Coke | Drink | 0.9 | L |
| 3 Juice | Drink | 0.5 | L |
| 4 Apple | Fruit | 0.99 | LB |
| 5 Avocado | Fruit | 1.49 | EA |
| 6 Pineapple | Fruit | 1.49 | LB |
| 7 Peach | Fruit | 0.88 | LB |
| 8 Orange | Fruit | 0.49 | LB |
| 9 Pork Spareribs | Meat | 1.99 | LB |
| 10 Drum stick | Meat | 0.99 | LB |
| 11 Fresh Salmon | Seafood | 7.99 | LB |
| 12 Razor Clam | Seafood | 3.99 | LB |

◆ 3.1 Aggregation on a computed column in a grouped table



Compute sales amount of each food and get a result set as follows:

| | Commodity | Type | UnitPrice | Unit | Quantity | Amount |
|----|----------------|---------|-----------|------|----------|---------------|
| 1 | Milk | Drink | 3.99 | GAL | 50 | 199.5 |
| 2 | Coke | Drink | 0.9 | L | 300 | 270.0 |
| 3 | Juice | Drink | 0.5 | L | 120 | 60.0 |
| 4 | Apple | Fruit | 0.99 | LB | 150.5 | 148.995 |
| 5 | Avocado | Fruit | 1.49 | EA | 77 | 114.73 |
| 6 | Pineapple | Fruit | 1.49 | LB | 30.99 | 46.1751 |
| 7 | Peach | Fruit | 0.88 | LB | 118.99 | 104.711199999 |
| 8 | Orange | Fruit | 0.49 | LB | 277.25 | 135.8525 |
| 9 | Pork Spareribs | Meat | 1.99 | LB | 58.45 | 116.3155 |
| 10 | Drum stick | Meat | 0.99 | LB | 35.88 | 35.5212 |
| 11 | Fresh Salmon | Seafood | 7.99 | LB | 25.66 | 205.0234 |
| 12 | Razor Clam | Seafood | 3.99 | LB | 33.89 | 135.2211 |

3.1 Aggregation on a computed column in a grouped table



Group rows by commodity type (Type field) and compute total sales amount of each type of food:

| | Commodity | Type | UnitPrice | Unit | Quantity |
|----|----------------|---------|-----------|------|----------|
| 1 | Milk | Drink | 3.99 | GAL | |
| 2 | Coke | Drink | 0.9 | L | |
| 3 | Juice | Drink | 0.5 | L | |
| 4 | Apple | Fruit | 0.99 | LB | |
| 5 | Avocado | Fruit | 1.49 | EA | |
| 6 | Pineapple | Fruit | 1.49 | LB | |
| 7 | Peach | Fruit | 0.88 | LB | |
| 8 | Orange | Fruit | 0.49 | LB | |
| 9 | Pork Spareribs | Meat | 1.99 | LB | |
| 10 | Drum stick | Meat | 0.99 | LB | |
| 11 | Fresh Salmon | Seafood | 7.99 | LB | |
| 12 | Razor Clam | Seafood | 3.99 | LB | |

Group dialog box details:

- Name: groupType
- Source table: food
- Aggregates: sum
- Grouping field: Type
- Source fields: Amount, Commodity, Quantity, Type, Unit, UnitPrice

Group by Type field

3.1 Aggregation on a computed column in a grouped table



Set the aggregate expression:

Group dialog box configuration:

- Name: groupType
- Source table: food
- Options: (empty)
- Aggregates: sum
- Source fields (Double click to select): Amount, Commodity, Quantity, Type, Unit, UnitPrice
- Grouping field: (empty)
- Aggregates table:

| Index | Aggregate expression | Alias |
|-------|----------------------|-------|
| 1 | sum(Amount) | Total |
- Keep the original order:
- Return the first row of each group:
- Group when the key changes:
- Parallel computation:
- Grouping key is sequence number:
- Discard groups with null key:
- Group when the key is true:
- Retain details:

Total sales amounts of each type of foods:

| | Type | Total |
|---|-----------|------------|
| 1 | Drink | 529.5 |
| 2 | Fruit | 550.4638 |
| 3 | Meat | 151.8367 |
| 4 | Seafood | 1003.87200 |
| 5 | Vegetable | 214.8515 |

3.1 Aggregation on a computed column in a grouped table



Group the GroupType table without the need of setting up grouping field, and compute the grand total:

Group dialog box configuration:

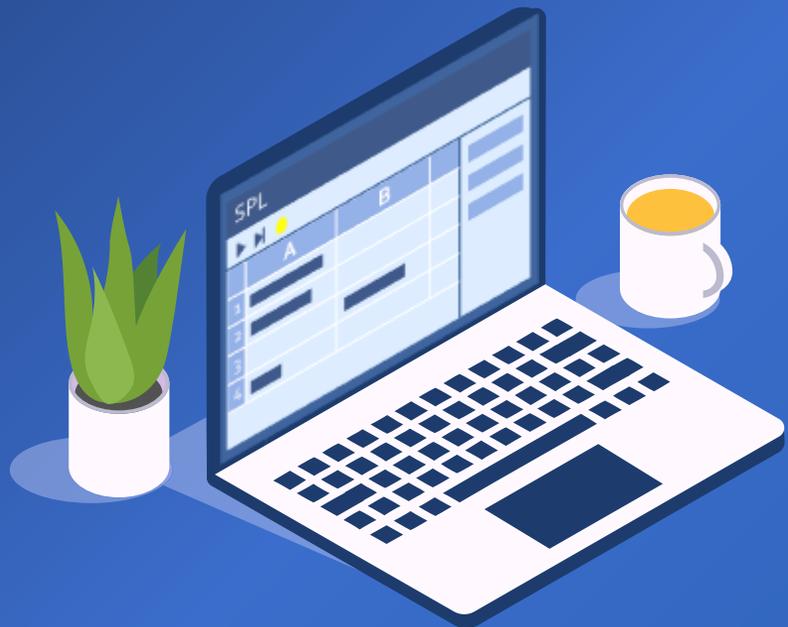
- Name: groupAll
- Source table: groupType
- Aggregates: sum
- Aggregate expression table:

| Index | Aggregate expression | Alias |
|-------|----------------------|------------|
| 1 | sum(Total) | GrandTotal |
- Source fields: Total, Type

The total sales amount of all foods:

| food | groupType | groupAll |
|------|-----------|--------------------|
| | | GrandTotal |
| 1 | | 2450.5240000000003 |

CONTENTS



Intragroup sorting on a grouped table

✦ 3.2 Intragroup sorting on a grouped table



Based on transaction records of stock 600038 in June, 2023, compute the growth rate of each date's closing price, and group rows by week and sort them by closing price in descending order for the convenience of viewing. Open table [600038.3.2.xlsx](#) and its data is as follows:

If a file name is a number, a prefix will by default added to form a legal identifier

| | code | date | open | close |
|----|--------|------------|------|-------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 |
| 2 | 600038 | 2023-06-05 | 5.1 | 4.75 |
| 3 | 600038 | 2023-06-06 | 4.91 | 4.63 |
| 4 | 600038 | 2023-06-07 | 4.28 | 4.25 |
| 5 | 600038 | 2023-06-08 | 4.51 | 4.39 |
| 6 | 600038 | 2023-06-09 | 4.43 | 4.81 |
| 7 | 600038 | 2023-06-12 | 4.63 | 4.92 |
| 8 | 600038 | 2023-06-13 | 4.69 | 4.49 |
| 9 | 600038 | 2023-06-14 | 4.53 | 4.84 |
| 10 | 600038 | 2023-06-15 | 5.61 | 5.2 |
| 11 | 600038 | 2023-06-16 | 5.22 | 5.35 |
| 12 | 600038 | 2023-06-19 | 5.5 | 5.25 |

✦ 3.2 Intragroup sorting on a grouped table



Sort rows by transaction date:

The screenshot shows a data table with columns: code, date, open, and close. The table contains 12 rows, all with the same 'code' value of 600038. The 'date' column shows two distinct dates: 2023-06-02 and 2023-06-05. A 'Sort' dialog box is open over the table, with the 'date' field selected as the sorting field in ascending order. The dialog also shows the source table name as '_600038' and the source fields as 'close', 'code', 'date', and 'open'.

| | code | date | open | close |
|----|--------|------------|------|-------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 |
| 2 | 600038 | 2023-06-05 | 5.1 | 4.75 |
| 3 | 600038 | | | |
| 4 | 600038 | | | |
| 5 | 600038 | | | |
| 6 | 600038 | | | |
| 7 | 600038 | | | |
| 8 | 600038 | | | |
| 9 | 600038 | | | |
| 10 | 600038 | | | |
| 11 | 600038 | | | |
| 12 | 600038 | | | |

Sort

Name: _600038 Source table: _600038

Locale: English Options: []

Sorting field: [+] [-] [↑] [↓] Source fields (Double click to select): close, code, date, open

| Index | Field | Ascending |
|-------|-------|-------------------------------------|
| 1 | date | <input checked="" type="checkbox"/> |

Parallel computation Place null values at the end

✦ 3.2 Intragroup sorting on a grouped table



Then set up a computed column to add to the table and compute growth rate of each closing price:

The screenshot shows a data table with columns: code, date, open, and close. The table contains 12 rows of data for code 600038. A dialog box titled "Computed column" is open, showing the configuration for a new column. The "Name" field is set to "_600038" and the "Source table" is "_600038". The "Computed column" section has a table with the following content:

| Index | Expression | Alias |
|-------|----------------------------|--------|
| 1 | if(##=1,0,close-close[-1]) | Growth |

A red arrow points from the expression in the dialog to a blue callout box containing the text: 表达式: if(##=1,0,close-close[-1])

✦ 3.2 Intragroup sorting on a grouped table



And get the following result set:

| | code | date | open | close | Growth |
|----|--------|------------|------|-------|----------------------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 | 0 |
| 2 | 600038 | 2023-06-05 | 5.1 | 4.75 | 0.25 |
| 3 | 600038 | 2023-06-06 | 4.91 | 4.63 | -0.12000000000000001 |
| 4 | 600038 | 2023-06-07 | 4.28 | 4.25 | -0.37999999999999999 |
| 5 | 600038 | 2023-06-08 | 4.51 | 4.39 | 0.13999999999999968 |
| 6 | 600038 | 2023-06-09 | 4.43 | 4.81 | 0.41999999999999993 |
| 7 | 600038 | 2023-06-12 | 4.63 | 4.92 | 0.110000000000000032 |
| 8 | 600038 | 2023-06-13 | 4.69 | 4.49 | -0.42999999999999997 |
| 9 | 600038 | 2023-06-14 | 4.53 | 4.84 | 0.34999999999999964 |
| 10 | 600038 | 2023-06-15 | 5.61 | 5.2 | 0.36000000000000003 |
| 11 | 600038 | 2023-06-16 | 5.22 | 5.35 | 0.14999999999999947 |
| 12 | 600038 | 2023-06-19 | 5.5 | 5.25 | -0.09999999999999964 |

✦ 3.2 Intragroup sorting on a grouped table



Group transaction records
by week:

Group dialog box configuration:

- Name: groupWeek
- Source table: _600038
- Options: i
- Aggregates: sum
- Grouping field: day@w(date)==2
- Source fields: Growth, close, code, date, open
- Group when the key is true:
- Retain details:

Select the
two options

✦ 3.2 Intragroup sorting on a grouped table



Below is result set of grouping records by week:

| | v_date_date | Details |
|---|-------------|---------------------------------|
| 1 | false | [[600038,2023-06-02,4.39, ...]] |
| 2 | true | [[600038,2023-06-05,5.1, |
| 3 | true | [[600038,2023-06-12,4.63, |
| 4 | true | [[600038,2023-06-19,5.5, |
| 5 | true | [[600038,2023-06-26,4.8, |

Open the details table of the 2nd row and sort it:

| | code | date | open | close | Growth |
|---|--------|------------|------|-------|----------------------|
| 1 | 600038 | 2023-06-05 | 5.1 | 4.75 | 0.25 |
| 2 | 600038 | 2023-06-06 | 4.91 | 4.63 | -0.12000000000000001 |
| 3 | 600038 | 2023-06-07 | 4.28 | 4.25 | -0.37999999999999999 |
| 4 | 600038 | 2023-06-08 | 4.51 | 4.39 | 0.13999999999999968 |
| 5 | 600038 | 2023-06-09 | 4.43 | 4.81 | 0.41999999999999993 |

3.2 Intragroup sorting on a grouped table



Sort records by closing price field close:

| | code | date | open | close | Growth |
|---|--------|------------|------|-------|--------|
| 1 | 600038 | 2023-06-05 | 5.1 | 4.75 | 0.25 |
| 2 | 600038 | | | | |
| 3 | 600038 | | | | |
| 4 | 600038 | | | | |
| 5 | 600038 | | | | |

| Index | Field | Ascending |
|-------|-------|--------------------------|
| 1 | close | <input type="checkbox"/> |

The sorting result set:

| | code | date | open | close | Growth |
|---|--------|------------|------|-------|----------------------|
| 1 | 600038 | 2023-06-09 | 4.43 | 4.81 | 0.41999999999999993 |
| 2 | 600038 | 2023-06-05 | 5.1 | 4.75 | 0.25 |
| 3 | 600038 | 2023-06-06 | 4.91 | 4.63 | -0.12000000000000001 |
| 4 | 600038 | 2023-06-08 | 4.51 | 4.39 | 0.139999999999999968 |
| 5 | 600038 | 2023-06-07 | 4.28 | 4.25 | -0.37999999999999999 |

Tips: Computations on one details table via the toolbar icons are also valid on all the other details table

3.2 Intragroup sorting on a grouped table



Concatenate all details records in GroupWeek table as a table sequence:

Below is the result set:

The screenshot shows a data tool interface with two windows. The left window is titled 'groupWeek.Details(2)' and shows a configuration for concatenating grouped data. The 'Name' field is set to 'AllWeekDescend' and the 'Detail field' is set to 'Details'. The right window is titled 'AllWeekDescend' and shows a table with columns: code, date, open, close, and Growth. The table contains 16 rows of data. A yellow callout bubble points to the 'close' column with the text 'Closing prices of each week are sorted in descending order'.

| | code | date | open | close | Growth |
|----|--------|------------|------|-------|----------------------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 | 0 |
| 2 | 600038 | 2023-06-09 | 4.43 | 4.81 | 0.41999999999999993 |
| 3 | 600038 | 2023-06-05 | 5.1 | 4.75 | 0.25 |
| 4 | 600038 | 2023-06-06 | 4.91 | 4.63 | -0.12000000000000001 |
| 5 | 600038 | 2023-06-08 | 4.51 | 4.39 | 0.13999999999999968 |
| 6 | 600038 | 2023-06-07 | 4.28 | 4.25 | -0.37999999999999999 |
| 7 | 600038 | 2023-06-16 | 5.22 | 5.35 | 0.14999999999999947 |
| 8 | 600038 | 2023-06-15 | 5.61 | 5.2 | 0.36000000000000003 |
| 9 | 600038 | 2023-06-12 | 4.63 | 4.92 | 0.110000000000000032 |
| 10 | 600038 | 2023-06-14 | 4.53 | 4.84 | 0.34999999999999964 |
| 11 | 600038 | 2023-06-13 | 4.69 | 4.49 | -0.42999999999999997 |
| 12 | 600038 | 2023-06-19 | 5.5 | 5.25 | -0.09999999999999964 |
| 13 | 600038 | 2023-06-20 | 4.71 | 4.86 | -0.38999999999999997 |
| 14 | 600038 | 2023-06-21 | 4.65 | 4.65 | -0.20999999999999996 |
| 15 | 600038 | 2023-06-23 | 4.51 | 4.55 | 0.21999999999999975 |
| 16 | 600038 | 2023-06-22 | 4.04 | 4.33 | -0.32000000000000003 |

CONTENTS



Intragroup filtering on a grouped table

✦ 3.3 Intragroup filtering on a grouped table



Below is table [Temperature.3.3.xlsx](#) that stores records of a region's average temperature in each month. The task is to group rows by year and select each year's months whose average temperatures are greater than the year's average.

| | ID | Year | Month | TempF |
|----|----|------|-------|-------|
| 1 | 1 | 2010 | Jan | 55.07 |
| 2 | 2 | 2010 | Feb | 53.88 |
| 3 | 3 | 2010 | Mar | 62.89 |
| 4 | 4 | 2010 | Apr | 67.86 |
| 5 | 5 | 2010 | May | 72.81 |
| 6 | 6 | 2010 | Jun | 81.53 |
| 7 | 7 | 2010 | Jul | 84.44 |
| 8 | 8 | 2010 | Aug | 82.29 |
| 9 | 9 | 2010 | Sep | 81.97 |
| 10 | 10 | 2010 | Oct | 68.94 |
| 11 | 11 | 2010 | Nov | 61.26 |
| 12 | 12 | 2010 | Dec | 53.49 |
| 13 | 13 | 2011 | Jan | 50.72 |
| 14 | 14 | 2011 | Feb | 54.67 |
| 15 | 15 | 2011 | Mar | 61.88 |

3.3 Intragroup filtering on a grouped table



Group rows by Year and retain detailed data:

The screenshot shows a data tool interface with a table titled "Temperature". The table has columns: ID, Year, Month, and TempF. The data rows are as follows:

| ID | Year | Month | TempF |
|----|------|-------|-------|
| 1 | 2010 | Jan | 55.07 |
| 2 | | | |
| 3 | | | |
| 4 | | | |
| 5 | | | |
| 6 | | | |
| 7 | | | |
| 8 | | | |
| 9 | | | |
| 10 | | | |
| 11 | | | |
| 12 | | | |
| 13 | | | |
| 14 | | | |
| 15 | | | |
| 16 | | | |

A "Group" dialog box is open, showing the following configuration:

- Name: groupYear
- Source table: Temperature
- Aggregates: sum
- Grouping field: Year
- Retain details:

Set average temperature aggregate value:

The close-up shows the "Aggregates" section of the dialog box with the following configuration:

- Aggregates: avg
- Grouping field: Year

A table below shows the aggregate expression and alias:

| Index | Aggregate expression | Alias |
|-------|----------------------|----------|
| 1 | avg(TempF) | avgTempF |

3.3 Intragroup filtering on a grouped table



Result set of grouping operation:

| | Year | avgTempF | Details |
|---|------|-------------------|---------------------------------|
| 1 | 2010 | 68.86916666666667 | [[1,2010,Jan, ...],[2,2010,Feb, |
| 2 | 2011 | 68.1125 | [[13,2011,Jan, |
| 3 | 2012 | 68.90416666666665 | [[25,2012,Jan, |
| 4 | 2013 | 68.86416666666666 | [[37,2013,Jan, |
| 5 | 2014 | 68.2825 | [[49,2014,Jan, |
| 6 | 2015 | 67.88416666666667 | [[61,2015,Jan, |

Open details table of the year 2010:

| | ID | Year | Month | TempF |
|----|----|------|-------|-------|
| 1 | 1 | 2010 | Jan | 55.07 |
| 2 | 2 | 2010 | Feb | 53.88 |
| 3 | 3 | 2010 | Mar | 62.89 |
| 4 | 4 | 2010 | Apr | 67.86 |
| 5 | 5 | 2010 | May | 72.81 |
| 6 | 6 | 2010 | Jun | 81.53 |
| 7 | 7 | 2010 | Jul | 84.44 |
| 8 | 8 | 2010 | Aug | 82.29 |
| 9 | 9 | 2010 | Sep | 81.97 |
| 10 | 10 | 2010 | Oct | 68.94 |
| 11 | 11 | 2010 | Nov | 61.26 |
| 12 | 12 | 2010 | Dec | 53.49 |

3.3 Intragroup filtering on a grouped table



Set the filtering condition:

The screenshot shows a data table with columns ID, Year, Month, and TempF. A filter dialog box is open over the table, showing the filter expression 'TempF > avgTempF' in a text field. The dialog also shows a list of fields and values for selection.

| ID | Year | Month | TempF |
|----|------|-------|-------|
| 1 | 2010 | Jan | 55.07 |

Filter dialog box details:

- Name: groupYear.Details(1)
- Source table: groupYear.Details(1)
- Filter expression: TempF > avgTempF
- Field (Double click to select): ID, Month, TempF, Year
- Value (Double click to select): avgTempF, 53.49, 53.88, 55.07, 61.26, 62.89, 67.86
- Operator: +, -, *, /, <, >, <=, >=, =, !=, (,)

Result set of filtering a details table:

| ID | Year | Month | TempF |
|----|------|-------|-------|
| 5 | 2010 | May | 72.81 |
| 6 | 2010 | Jun | 81.53 |
| 7 | 2010 | Jul | 84.44 |
| 8 | 2010 | Aug | 82.29 |
| 9 | 2010 | Sep | 81.97 |
| 10 | 2010 | Oct | 68.94 |

3.3 Intragroup filtering on a grouped table



Concatenate all detailed records in groupYear table:

The screenshot shows a data tool interface with a table named 'groupYear'. The table has columns 'Year' and 'avgTempF'. A configuration dialog is open, titled 'Concatenate grouped result'. The dialog has the following fields:

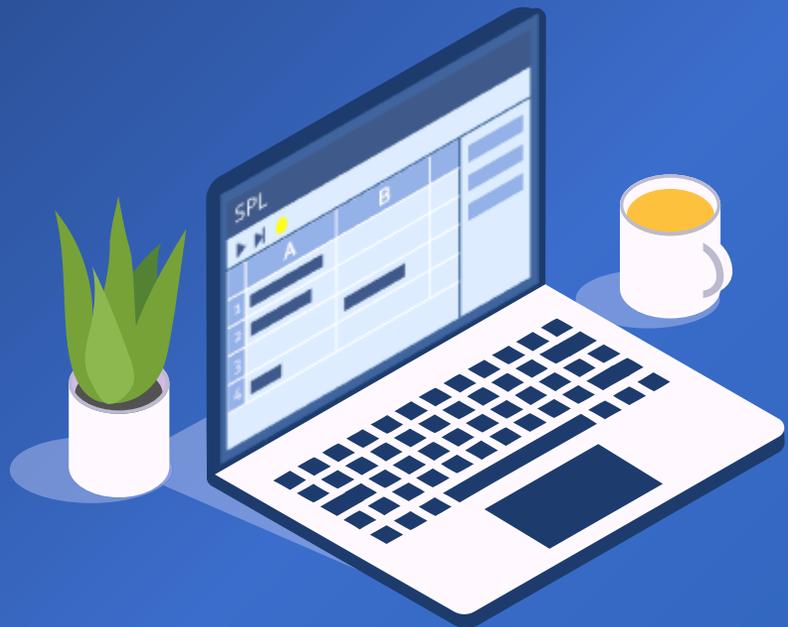
- Name: AllOverAvgMonth
- Source table: groupYear
- Detail field: Details
- Options: (empty)
- Recursive computation:

The final filtering result set:

The screenshot shows a data tool interface with a table named 'AllOverAvgMonth'. The table has columns 'ID', 'Year', 'Month', and 'TempF'. The table contains 15 rows of data:

| ID | Year | Month | TempF |
|----|------|-------|-------|
| 1 | 2010 | May | 72.81 |
| 2 | 2010 | Jun | 81.53 |
| 3 | 2010 | Jul | 84.44 |
| 4 | 2010 | Aug | 82.29 |
| 5 | 2010 | Sep | 81.97 |
| 6 | 2010 | Oct | 68.94 |
| 7 | 2011 | May | 71.71 |
| 8 | 2011 | Jun | 79.33 |
| 9 | 2011 | Jul | 85.37 |
| 10 | 2011 | Aug | 84.57 |
| 11 | 2011 | Sep | 76.79 |
| 12 | 2011 | Oct | 73.17 |
| 13 | 2012 | May | 72.25 |
| 14 | 2012 | Jun | 78.84 |
| 15 | 2012 | Jul | 78.21 |

CONTENTS



**Retain the first row of
each group**

✦ 3.4 Retain the first row of each group



Below are records of users' accesses to a website sections. We need to find record of each user's last access to the website. Open [UserVisit.3.4.xlsx](#) as follows:

| | ID | Section | AccessTime |
|----|----------|---------|------------|
| 1 | Nancy | Movie | 09:38:25 |
| 2 | Judy | Movie | 10:59:27 |
| 3 | Margaret | Movie | 10:59:50 |
| 4 | Paul | Movie | 12:03:32 |
| 5 | Nancy | Movie | 12:23:00 |
| 6 | Paul | Movie | 17:58:59 |
| 7 | Margaret | Movie | 18:35:40 |
| 8 | Nancy | Music | 12:02:33 |
| 9 | Judy | Music | 12:02:34 |
| 10 | Margaret | Music | 12:55:55 |
| 11 | Margaret | Music | 13:30:00 |
| 12 | Judy | Music | 13:44:28 |
| 13 | Nancy | Music | 14:55:08 |
| 14 | Nancy | Music | 16:00:55 |
| 15 | Paul | Admin | 07:23:55 |

◆ 3.4 Retain the first row of each group



Sort UserVisit:

The screenshot shows a data table with columns ID, Section, and AccessTime. A 'Sort' dialog box is open, showing the source table 'UserVisit' and the sorting field 'ID' in ascending order. The 'Sort' dialog is highlighted with a red box.

And get the following result:

| | ID | Section | AccessTime |
|----|----------|---------|------------|
| 1 | Judy | Music | 13:44:28 |
| 2 | Judy | Music | 12:02:34 |
| 3 | Judy | Movie | 10:59:27 |
| 4 | Margaret | Movie | 18:35:40 |
| 5 | Margaret | Music | 13:30:00 |
| 6 | Margaret | Music | 12:55:55 |
| 7 | Margaret | Movie | 10:59:50 |
| 8 | Nancy | Music | 16:00:55 |
| 9 | Nancy | Music | 14:55:08 |
| 10 | Nancy | Movie | 12:23:00 |
| 11 | Nancy | Music | 12:02:33 |
| 12 | Nancy | Movie | 09:38:25 |
| 13 | Paul | Admin | 18:00:35 |
| 14 | Paul | Movie | 17:58:59 |
| 15 | Paul | Admin | 13:00:06 |

3.4 Retain the first row of each group



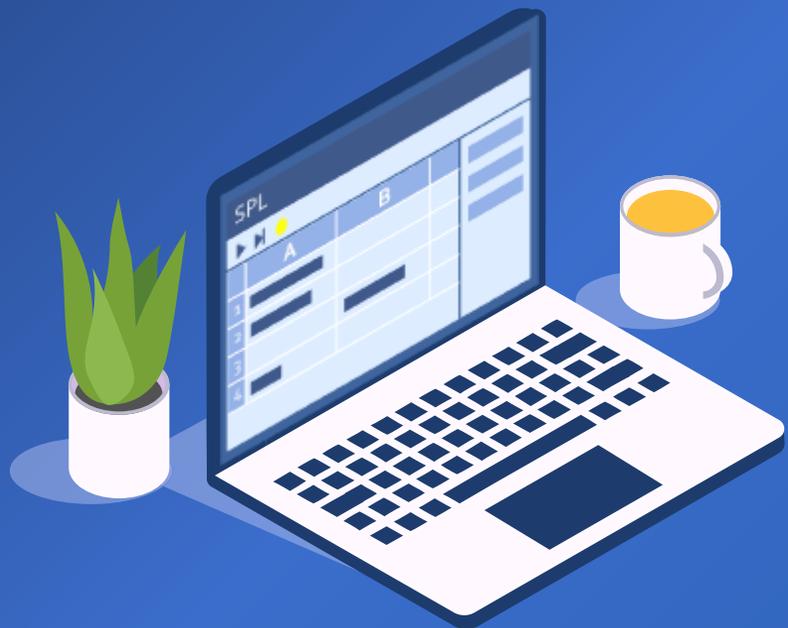
Perform the following filtering on UserVisit:

The screenshot shows a data table with columns ID, Section, and AccessTime. A filter dialog box is open, showing the filter expression `ID!=ID[-1]`. A red circle highlights the filter icon in the toolbar, and a red arrow points from the expression in the dialog to the text 'Expression: ID!=ID[-1]'. The dialog also shows a field list with 'AccessTime' selected.

And get the last access record:

| | ID | Section | AccessTime |
|---|----------|---------|------------|
| 1 | Judy | Music | 13:44:28 |
| 2 | Margaret | Movie | 18:35:40 |
| 3 | Nancy | Music | 16:00:55 |
| 4 | Paul | Admin | 18:00:35 |

CONTENTS



Grouping & aggregation after filtering

✦ 3.5 Grouping & aggregation after filtering



We have sales records of various wines, and we want to find the two locations of production having the largest sales for the two most popular types of wine. Open

Wine.3.5.xlsx as follows:

| | Name | Type | Production | Amount |
|----|-------------|----------|------------|--------|
| 1 | Baileys | Cordials | Ireland | 43400 |
| 2 | Ballantines | Whisky | Scotland | 23700 |
| 3 | Bombay | Gin | England | 6080 |
| 4 | Chatelle | Brandy | France | 6650 |
| 5 | Chivas | Whisky | Scotland | 43000 |
| 6 | Conjure | Brandy | France | 5830 |
| 7 | Cougar | Whisky | Australia | 2770 |
| 8 | Dewar's | Whisky | Scotland | 2150 |
| 9 | Drambuie | Cordials | Scotland | 8400 |
| 10 | Gilbeys | Gin | England | 2780 |
| 11 | Gordons | Gin | England | 28700 |
| 12 | Hennessy | Brandy | France | 7830 |
| 13 | Hine | Brandy | France | 6870 |
| 14 | Jim Beam | Whisky | American | 6690 |

3.5 Grouping & aggregation after filtering



Group the Wine table by Type and retain detailed data:

Set aggregate expression:

| Name | Type | Production | Amount |
|---------------|----------|------------|--------|
| 1 Baileys | Cordials | Ireland | 43400 |
| 2 Ballant... | | | |
| 3 Bomba... | | | |
| 4 Chatell... | | | |
| 5 Chivas... | | | |
| 6 Conjure... | | | |
| 7 Cougar... | | | |
| 8 Dewar's... | | | |
| 9 Drambu... | | | |
| 10 Gilbeys... | | | |
| 11 Gordor... | | | |
| 12 Hennes... | | | |
| 13 Hine | | | |
| 14 Jim Bea... | | | |
| 15 Kahlua... | | | |
| 16 Malibu... | | | |

Group Dialog 1:

- Name: groupType
- Source table: Wine
- Options: (empty)
- Aggregates: sum
- Grouping field: Type
- Aggregates list: Amount
- Retain details:

Group Dialog 2:

- Name: groupType
- Source table: Wine
- Options: (empty)
- Aggregates: sum
- Grouping field: (empty)
- Aggregates list:

| Index | Aggregate expression | Alias |
|-------|----------------------|-------|
| 1 | sum(Amount) | Total |
- Retain details:

3.5 Grouping & aggregation after filtering



Then sort rows by Total:

Get grouping result as follows:

| | Type | Total | Details |
|---|----------|--------|--------------------------------|
| 1 | Brandy | 28530 | [[Chatelle,Brandy,France, |
| 2 | Cordials | 73800 | [[Baileys,Cordials,Ireland, |
| 3 | Gin | 62060 | [[Bombay,Gin,England, |
| 4 | Rum | 3060 | [[Malibu,Rum,England, ...]] |
| 5 | Whisky | 105730 | [[Ballantines,Whisky,Scotland, |

Sort dialog box configuration:

- Name: groupType
- Source table: groupType
- Locale: English
- Sorting field: Total (highlighted in red)
- Ascending:

| Index | Field | Ascending |
|-------|-------|--------------------------|
| 1 | Total | <input type="checkbox"/> |

3.5 Grouping & aggregation after filtering



Get the top 2 rows in terms of total sales amount:

Sorting result set:

| | Type | Total | Details |
|---|----------|--------|--------------------------------|
| 1 | Whisky | 105730 | [[Ballantines,Whisky,Scotland, |
| 2 | Cordials | 73800 | [[Baileys,Cordials,Ireland, |
| 3 | Gin | 62060 | [[Bombay,Gin,England, |
| 4 | Brandy | 28530 | [[Chatelle,Brandy,France, |
| 5 | Rum | 3060 | [[Malibu,Rum,England, ...]] |

The screenshot shows a data tool interface with a table and a filter dialog box. The table has columns: Type, Total, and Details. The filter dialog box is titled 'Filter' and has the following fields:

- Name: Top2
- Source table: groupType
- Filter expression: #<3

The filter dialog box also has a table for defining the filter expression:

| Field (Double click to select) | Value (Double click to select) | Operator |
|--------------------------------|--------------------------------|----------|
| Details | 105730 | + - |
| Total | 28530 | * / |
| Type | 3060 | < > |
| | 62060 | <= >= |
| | 73800 | = != |

✦ 3.5 Grouping & aggregation after filtering



Filtering result:

| | Type | Total | Details |
|---|----------|--------|--------------------------------|
| 1 | Whisky | 105730 | [[Ballantines,Whisky,Scotland, |
| 2 | Cordials | 73800 | [[Baileys,Cordials,Ireland, |

Concatenate detailed data and name the new table Top2:

| | Type | Total | Details |
|---|----------------------------|--------|--------------------------------|
| 1 | Whisky | 105730 | [[Ballantines,Whisky,Scotland, |
| 2 | Concatenate grouped result | | |

Name **Source table**

Detail field **Options**

Recursive computation

3.5 Grouping & aggregation after filtering



Group Top2 table by Production:

Concatenation result Top2:

| | Name | Type | Production | Amount |
|----|-------------|----------|------------|--------|
| 1 | Ballantines | Whisky | Scotland | 23700 |
| 2 | Chivas | Whisky | Scotland | 43000 |
| 3 | Cougar | Whisky | Australia | 27000 |
| 4 | Dewar's | Whisky | Scotland | 21000 |
| 5 | Jim Beam | Whisky | American | 66000 |
| 6 | McKenna | Whisky | American | 76000 |
| 7 | Teachers | Whisky | Scotland | 143000 |
| 8 | Glenlivet | Whisky | Scotland | 58000 |
| 9 | Baileys | Cordials | Ireland | 434000 |
| 10 | Drambuie | Cordials | Scotland | 84000 |
| 11 | Kahlua | Cordials | Mexico | 220000 |

Wine groupType Top2

Group dialog box:

- Name: groupProduction
- Source table: Top2
- Aggregates: sum
- Grouping field: Production
- Aggregate expression: sum(Amount)
- Alias: Total
- Retain details:

| Index | Field expression |
|-------|------------------|
| 1 | Production |

| Index | Aggregate expression | Alias |
|-------|----------------------|-------|
| 1 | sum(Amount) | Total |

Options:

- Keep the original order
- Return the first row of each group
- Group when the key changes
- Parallel computation
- Grouping key is sequence number
- Discard groups with null key
- Group when the key is true

3.5 Grouping & aggregation after filtering



Then sort rows by Total:

Grouping result set:

| | Production | Total | Details |
|---|------------|-------|-----------------------------------|
| 1 | American | 14290 | [[Jim Beam,Whisky,American, |
| 2 | Australia | 2770 | [[Cougar,Whisky,Australia, ...]] |
| 3 | Mexico | 22000 | [[Kahlua,Cordials,Mexico, ...]] |
| 4 | Scotland | 97070 | [[Ballantines,Whisky,Scotland, |
| 5 | Ireland | 43400 | [[Baileys,Cordials,Ireland, ...]] |

Wine groupType Top2 groupProduction

Sort

Name: groupProduction Source table: groupProduction

Locale: English Options: [OK] [Cancel]

| Index | Field | Ascending |
|-------|-------|-------------------------------------|
| 1 | Total | <input checked="" type="checkbox"/> |

Source fields (Double click to select):

- Details
- Production
- Total

Parallel computation Place null values at the end

✦ 3.5 Grouping & aggregation after filtering



And get the Top2 in terms of Production:

| | Production | Total | Details |
|---|------------|-------|--------------------------------------|
| 1 | Scotland | 97070 | [[Ballantines,Whisky,Scotland, ...]] |
| 2 | Ireland | 43400 | [[Baileys,Cordials,Ireland, ...]] |
| 3 | Mexico | 22000 | [[Kahlua,Cordials,Mexico, ...]] |
| 4 | American | 14290 | [[Jim Beam,Whisky,American, ...]] |
| 5 | Australia | 2770 | [[Cougar,Whisky,Australia, ...]] |

CONTENTS



Compute link relative in a multilayer-structure table

✦ 3.6 Compute link relative in a multilayer-structure table



The table below stores population data, and we want to find the three states having the highest population growth rates in the year 2010. Below is data in [Populations.3.6.xlsx](#) :

The population data is collected every one decade, and there are three years of data

| | State | Year | Population |
|----|----------------|------|------------|
| 1 | Texas | 2010 | 25145561 |
| 2 | Texas | 2000 | 20851820 |
| 3 | Texas | 1990 | 16987000 |
| 4 | North Carolina | 2010 | 9535483 |
| 5 | North Carolina | 2000 | 8049313 |
| 6 | North Carolina | 1990 | 6629000 |
| 7 | Arizona | 2010 | 6392017 |
| 8 | Arizona | 2000 | 5130632 |
| 9 | Arizona | 1990 | 3685000 |
| 10 | Utah | 2010 | 2763885 |
| 11 | Nevada | 2010 | 2700551 |
| 12 | Utah | 2000 | 2233169 |
| 13 | Nevada | 2000 | 1998257 |
| 14 | Utah | 1990 | 1723000 |

3.6 Compute link relative in a multilayer-structure table



To compute growth rate of 2010 only, filter away population data of 1990:

Populations

ABC 123       

| | State | Year | Population |
|---|-------|------|------------|
| 1 | Texas | 2010 | 25145561 |

2  Filter

3 Name Source table

4

5 Filter expression:

6 1

7

8

| Field (Double click to select) | Value (Double click to select) | Operator |
|--------------------------------|--------------------------------|--|
| Population | 1990 | <input type="button" value="+"/> <input type="button" value="-"/> |
| State | 2000 | <input type="button" value="*"/> <input type="button" value="/"/> |
| Year | 2010 | <input type="button" value="<"/> <input type="button" value=">"/> |
| | | <input type="button" value="<="/> <input "="" type="button" value=">="/> |
| | | <input type="button" value="="/> <input type="button" value!=""/> |

9

10

11

12

13

14

Filtering result set:

Populations

ABC 123       

| | State | Year | Population |
|----|----------------|------|------------|
| 1 | Texas | 2010 | 25145561 |
| 2 | Texas | 2000 | 20851820 |
| 3 | North Carolina | 2010 | 9535483 |
| 4 | North Carolina | 2000 | 8049313 |
| 5 | Arizona | 2010 | 6392017 |
| 6 | Arizona | 2000 | 5130632 |
| 7 | Utah | 2010 | 2763885 |
| 8 | Nevada | 2010 | 2700551 |
| 9 | Utah | 2000 | 2233169 |
| 10 | Nevada | 2000 | 1998257 |
| 11 | Idaho | 2010 | 1567582 |
| 12 | Idaho | 2000 | 1293953 |

3.6 Compute link relative in a multilayer-structure table



Sort rows by State and Year in ascending order:

The screenshot shows a 'Populations' table with the following data:

| | State | Year | Population |
|---|-------|------|------------|
| 1 | Texas | 2010 | 25145561 |

The 'Sort' dialog box is open, showing the following configuration:

- Name: Populations
- Source table: Populations
- Locale: English
- Sorting field: State (Index 1), Year (Index 2)
- Ascending: checked for both State and Year

Sorting result set:

The screenshot shows the 'Populations' table after sorting by State and Year in ascending order:

| | State | Year | Population |
|----|----------------|------|------------|
| 1 | Arizona | 2000 | 5130632 |
| 2 | Arizona | 2010 | 6392017 |
| 3 | Idaho | 2000 | 1293953 |
| 4 | Idaho | 2010 | 1567582 |
| 5 | Nevada | 2000 | 1998257 |
| 6 | Nevada | 2010 | 2700551 |
| 7 | North Carolina | 2000 | 8049313 |
| 8 | North Carolina | 2010 | 9535483 |
| 9 | Texas | 2000 | 20851820 |
| 10 | Texas | 2010 | 25145561 |
| 11 | Utah | 2000 | 2233169 |
| 12 | Utah | 2010 | 2763885 |

3.6 Compute link relative in a multilayer-structure table



Compute population growth rate:

Populations

| | State | Year | Population |
|---|---------|------|------------|
| 1 | Arizona | 2000 | 5130632 |

Computed column

Name: Populations Source table: Populations

Options:

Computed column: + - ↑ ↓

| Index | Expression | Alias |
|-------|--|------------|
| 1 | if(State==State[-1],(Popula GrowthRate | GrowthRate |

Source fields (Double click to select): Population, State, Year

Parallel computation

And get this result set:

Populations

| | State | Year | Population | GrowthRate |
|----|----------------|------|------------|---------------------|
| 1 | Arizona | 2000 | 5130632 | (null) |
| 2 | Arizona | 2010 | 6392017 | 0.24585372718214832 |
| 3 | Idaho | 2000 | 1293953 | (null) |
| 4 | Idaho | 2010 | 1567582 | 0.2114674953417937 |
| 5 | Nevada | 2000 | 1998257 | (null) |
| 6 | Nevada | 2010 | 2700551 | 0.35145329154358024 |
| 7 | North Carolina | 2000 | 8049313 | (null) |
| 8 | North Carolina | 2010 | 9535483 | 0.18463314819538015 |
| 9 | Texas | 2000 | 20851820 | (null) |
| 10 | Texas | 2010 | 25145561 | 0.20591684562786366 |
| 11 | Texas | 2000 | 2233169 | (null) |
| 12 | Utah | 2010 | 2763885 | 0.23765151674593368 |

Expression: `if(State==State[-1],(Population-Population[-1])/Population[-1],null)`

3.6 Compute link relative in a multilayer-structure table



Sort Populations table by GrowthRate:

Populations

| | State | Year | Population | GrowthRate |
|---|---------|------|------------|------------|
| 1 | Arizona | 2000 | 5130632 | (null) |

Sort

Name: Populations, Source table: Populations, Locale: English

Sorting field: GrowthRate

| Index | Field | Ascending |
|-------|------------|--------------------------|
| 1 | GrowthRate | <input type="checkbox"/> |

Source fields (Double click to select): GrowthRate, Population, State, Year

Sorting result set:

Populations

| | State | Year | Population | GrowthRate |
|----|----------------|------|------------|---------------------|
| 1 | Nevada | 2010 | 2700551 | 0.35145329154358024 |
| 2 | Arizona | 2010 | 6392017 | 0.24585372718214832 |
| 3 | Utah | 2010 | 2763885 | 0.23765151674593368 |
| 4 | Idaho | 2010 | 1567582 | 0.2114674953417937 |
| 5 | Texas | 2010 | 25145561 | 0.20591684562786366 |
| 6 | North Carolina | 2010 | 9535483 | 0.18463314819538015 |
| 7 | Arizona | 2000 | 5130632 | (null) |
| 8 | Idaho | 2000 | 1293953 | (null) |
| 9 | Nevada | 2000 | 1998257 | (null) |
| 10 | North Carolina | 2000 | 8049313 | (null) |
| 11 | Texas | 2000 | 20851820 | (null) |
| 12 | Utah | 2000 | 2233169 | (null) |

The three states having the highest growth rates

CONTENTS



Comparison between multiple tables

✦ 3.7 Comparison between multiple tables



Here are top 15 universities released by ARWU in 2010 and 2009. We want to find how many places each university has risen in the 2010 list compared to the previous year. Below are two sheet of data in [ARWU.3.7.xlsx](#) :

| | Y2009 | Y2010 | |
|----|-------|-------------------------------|---------|
| | Rank | University | Country |
| 1 | 1 | Harvard University | USA |
| 2 | 2 | University of | USA |
| 3 | 3 | Stanford University | USA |
| 4 | 4 | Massachusetts Institute of | USA |
| 5 | 5 | University of Cambridge | GBR |
| 6 | 6 | California Institute of | USA |
| 7 | 7 | Princeton University | USA |
| 8 | 8 | Columbia University | USA |
| 9 | 9 | University of Chicago | USA |
| 10 | 10 | University of Oxford | GBR |
| 11 | 11 | Yale University | USA |
| 12 | 12 | Cornell University | USA |
| 13 | 13 | University of California, Los | USA |
| 14 | 14 | University of California, San | USA |

3.7 Comparison between multiple tables



Join Y2010 and Y2009 as a wide table:

Y2009 Y2010

| | Rank | University | Country |
|---|------|--------------------|---------|
| 1 | 1 | Harvard University | USA |

Join

Name: **JoinUniversity** Source table: Y2010

Join type: Inner join Left join

Target table: Y2009

| Index | Target table | Select |
|-------|--------------|-------------------------------------|
| 1 | Y2009 | <input checked="" type="checkbox"/> |

Set the join field:

| Index | Y2010 | Y2009 |
|-------|------------|------------|
| 1 | University | University |

Set to-be-selected fields:

| Index | Table | Field | Select | Alias |
|-------|-------|------------|-------------------------------------|--------|
| 1 | Y2010 | Rank | <input checked="" type="checkbox"/> | Rank10 |
| 2 | Y2010 | University | <input checked="" type="checkbox"/> | |
| 3 | Y2010 | Country | <input checked="" type="checkbox"/> | |
| 4 | Y2009 | Rank | <input checked="" type="checkbox"/> | Rank09 |
| 5 | Y2009 | University | <input type="checkbox"/> | |
| 6 | Y2009 | Country | <input type="checkbox"/> | |

3.7 Comparison between multiple tables



Joining result set:

Sort rows by Rank10 in ascending order:

| | Rank10 | University | Country |
|----|--------|-------------------------------|---------|
| 1 | 2 | University of | USA |
| 2 | 3 | Stanford University | USA |
| 3 | 4 | Massachusetts Institute of | USA |
| 4 | 12 | Cornell University | USA |
| 5 | 6 | California Institute of | USA |
| 6 | 8 | Columbia University | USA |
| 7 | 11 | Yale University | USA |
| 8 | 1 | Harvard University | USA |
| 9 | 9 | University of Chicago | USA |
| 10 | 14 | University of California, San | USA |
| 11 | 10 | University of Oxford | GBR |
| 12 | 5 | University of Cambridge | GBR |
| 13 | 7 | Princeton University | USA |
| 14 | 13 | University of California, Los | USA |
| 15 | 15 | University of Pennsylvania | USA |

| | Rank10 | University | Country | Rank09 |
|----|--------|---------------|---------|--------|
| 1 | 2 | University of | USA | 3 |
| 2 | | | | |
| 3 | | | | |
| 4 | | | | |
| 5 | | | | |
| 6 | | | | |
| 7 | | | | |
| 8 | | | | |
| 9 | | | | |
| 10 | | | | |
| 11 | | | | |
| 12 | | | | |
| 13 | | | | |
| 14 | | | | |
| 15 | | | | |

Sort dialog box for JoinUniversity table. The sorting field is Rank10, sorted in ascending order. The dialog also shows source fields: Country, Rank09, Rank10, and University.

3.7 Comparison between multiple tables



Sorting result set:

| | Rank10 | University | Country | Rank09 |
|----|--------|-------------------------------|---------|--------|
| 1 | 1 | Harvard University | USA | 1 |
| 2 | 2 | University of | USA | 3 |
| 3 | 3 | Stanford University | USA | 2 |
| 4 | 4 | Massachusetts Institute of | USA | 5 |
| 5 | 5 | University of Cambridge | GBR | 4 |
| 6 | 6 | California Institute of | USA | 6 |
| 7 | 7 | Princeton University | USA | 8 |
| 8 | 8 | Columbia University | USA | 7 |
| 9 | 9 | University of Chicago | USA | 9 |
| 10 | 10 | University of Oxford | GBR | 10 |
| 11 | 11 | Yale University | USA | 11 |
| 12 | 12 | Cornell University | USA | 12 |
| 13 | 13 | University of California, Los | USA | 13 |
| 14 | 14 | University of California, San | USA | 14 |
| 15 | 15 | University of Pennsylvania | USA | 15 |

Count how many places each university has risen in 2010:

Y2009 Y2010 JoinUniversity

| | Rank10 | University | Country | Rank09 |
|----|---|--------------------|---------|--------|
| 1 | 1 | Harvard University | USA | 1 |
| 2 | Computed column | | | |
| 3 | Name: JoinUniversity Source table: JoinUniversity | | | |
| 4 | Options | | | |
| 5 | Computed column: + - ↑ ↓ | | | |
| 6 | Source fields (Double click to select) | | | |
| 7 | Country | | | |
| 8 | Rank09 | | | |
| 9 | Rank10 | | | |
| 10 | University | | | |
| 11 | | | | |
| 12 | | | | |
| 13 | | | | |
| 14 | | | | |
| 15 | | | | |

Parallel computation

Computed Column Configuration:

| Index | Expression | Alias |
|-------|---------------|----------|
| 1 | Rank09-Rank10 | Increase |

✦ 3.7 Comparison between multiple tables



Ranking changes result:

| | Rank10 | University | Country | Rank09 | Increase |
|----|--------|-------------------------------|---------|--------|----------|
| 1 | 1 | Harvard University | USA | 1 | 0 |
| 2 | 2 | University of | USA | 3 | 1 |
| 3 | 3 | Stanford University | USA | 2 | -1 |
| 4 | 4 | Massachusetts Institute of | USA | 5 | 1 |
| 5 | 5 | University of Cambridge | GBR | 4 | -1 |
| 6 | 6 | California Institute of | USA | 6 | 0 |
| 7 | 7 | Princeton University | USA | 8 | 1 |
| 8 | 8 | Columbia University | USA | 7 | -1 |
| 9 | 9 | University of Chicago | USA | 9 | 0 |
| 10 | 10 | University of Oxford | GBR | 10 | 0 |
| 11 | 11 | Yale University | USA | 11 | 0 |
| 12 | 12 | Cornell University | USA | 12 | 0 |
| 13 | 13 | University of California, Los | USA | 13 | 0 |
| 14 | 14 | University of California, San | USA | 14 | 0 |
| 15 | 15 | University of Pennsylvania | USA | 15 | 0 |

SPL WIN Course

Chapter 4

Comprehensive use cases

SPL WIN



CONTENTS



4.1 Get stock transaction records meeting the specified condition

4.2 Compute LRR and YOY based on car sales table

4.3 Associate worksheets and compute salaries

4.4 Find salespeople who rank top5 every month

CONTENTS



**Get stock transaction records
meeting the specified condition**

◆ 4.1 Get stock transaction records meeting the specified condition



Get stock transaction records where closing prices rise for at least 5 trading dates consecutively. Open [Stock.4.1.xlsx](#) and there are the following records:

| | code | date | open | close |
|----|--------|------------|--------|--------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 |
| 2 | 603818 | 2023-06-02 | 215.93 | 235 |
| 3 | 301500 | 2023-06-02 | 71 | 67.8 |
| 4 | 320688 | 2023-06-02 | 33 | 32 |
| 5 | 600788 | 2023-06-02 | 105.62 | 102.5 |
| 6 | 350880 | 2023-06-02 | 79.14 | 80 |
| 7 | 600038 | 2023-06-05 | 5.1 | 4.75 |
| 8 | 603818 | 2023-06-05 | 253.08 | 244.84 |
| 9 | 301500 | 2023-06-05 | 79.35 | 72.61 |
| 10 | 320688 | 2023-06-05 | 28.37 | 30.23 |
| 11 | 600788 | 2023-06-05 | 123.17 | 112.32 |
| 12 | 350880 | 2023-06-05 | 85.9 | 80.75 |
| 13 | 600038 | 2023-06-06 | 4.91 | 4.63 |
| 14 | 603818 | 2023-06-06 | 247.67 | 230.96 |
| 15 | 301500 | 2023-06-06 | 75.8 | 77.65 |

4.1 Get stock transaction records meeting the specified condition



Sort rows by stock code and date:

The 'Sort' dialog box is configured as follows:

- Name: Stock
- Source table: Stock
- Locale: English
- Options: (empty)
- Sorting field: code (Index 1), date (Index 2)
- Ascending: checked for both code and date

| | code | date | open | close |
|---|--------|------------|------|-------|
| 1 | 600038 | 2023-06-02 | 4.39 | 4.5 |

Sorting result set:

| | code | date | open | close |
|----|--------|------------|-------|-------|
| 1 | 301500 | 2023-06-02 | 71 | 67.8 |
| 2 | 301500 | 2023-06-05 | 79.35 | 72.61 |
| 3 | 301500 | 2023-06-06 | 75.8 | 77.65 |
| 4 | 301500 | 2023-06-07 | 74.63 | 72.55 |
| 5 | 301500 | 2023-06-08 | 76.1 | 74.97 |
| 6 | 301500 | 2023-06-09 | 65.23 | 68.22 |
| 7 | 301500 | 2023-06-12 | 70.4 | 75 |
| 8 | 301500 | 2023-06-13 | 82.69 | 81.22 |
| 9 | 301500 | 2023-06-14 | 76.52 | 80.28 |
| 10 | 301500 | 2023-06-15 | 82.13 | 79.78 |
| 11 | 301500 | 2023-06-16 | 88.32 | 85.75 |
| 12 | 301500 | 2023-06-19 | 98.36 | 91.05 |
| 13 | 301500 | 2023-06-20 | 86.87 | 82.27 |
| 14 | 301500 | 2023-06-21 | 81.17 | 85.11 |
| 15 | 301500 | 2023-06-22 | 88.65 | 93.45 |
| 16 | 301500 | 2023-06-23 | 83.48 | 84.77 |

4.1 Get stock transaction records meeting the specified condition



Perform order-based grouping by stock code:

Group

Name: **groupCode** Source table: Stock

Options: i

Aggregates: sum

Source fields (Double click to select): close, code, date, open

| Index | Field expression |
|-------|--|
| 1 | code!=code[-1] close<close[-1]:result |

Expression: code!=code[-1] || close<close[-1]:result

Return the first row of each group Discard groups with null key

Group when the key changes Group when the key is true

Parallel computation Retain details

Grouping result set:

| | result | Details |
|----|--------|-------------------------------|
| 1 | true | [[301500,2023-06-02,71, |
| 2 | false | [[301500,2023-06-07,74.63, |
| 3 | true | [[301500,2023-06-09,65.23, |
| 4 | false | [[301500,2023-06-14,76.52, |
| 5 | true | [[301500,2023-06-15,82.13, |
| 6 | false | [[301500,2023-06-20,86.87, |
| 7 | false | [[301500,2023-06-23,83.48, |
| 8 | false | [[301500,2023-06-28,91.01, |
| 9 | true | [[301500,2023-06-29,85.02, |
| 10 | true | [[301500,2023-06-30,82.69, |
| 11 | true | [[301500,2023-07-03,77.52, |
| 12 | false | [[301500,2023-07-05,78, ...]] |
| 13 | true | [[301500,2023-07-06,75.12, |
| 14 | true | [[301500,2023-07-07,70.11, |
| 15 | false | [[301500,2023-07-11,77.93, |

4.1 Get stock transaction records meeting the specified condition



Perform filtering to get groups where the stock rises for consecutive 5 trading dates:

The screenshot shows a data tool interface with a toolbar containing a filter icon (circled in red). A dialog box titled 'Filter' is open, showing the filter expression 'Details.len()>=5' highlighted in yellow. The dialog also shows the source table as 'groupCode' and a list of fields including 'Details' and 'result'.

Filtering result set:

| | result | Details |
|---|--------|-----------------------------|
| 1 | true | [[301500,2023-07-21,69.6, |
| 2 | true | [[320688,2023-06-29,17.65, |
| 3 | true | [[320688,2023-07-21,23.65, |
| 4 | false | [[350880,2023-07-25,65.09, |
| 5 | true | [[600788,2023-06-19,97.83, |
| 6 | true | [[600788,2023-06-29,111.31, |
| 7 | true | [[600788,2023-07-19,138.42, |
| 8 | true | [[603818,2023-07-20,159.36, |

4.1 Get stock transaction records meeting the specified condition



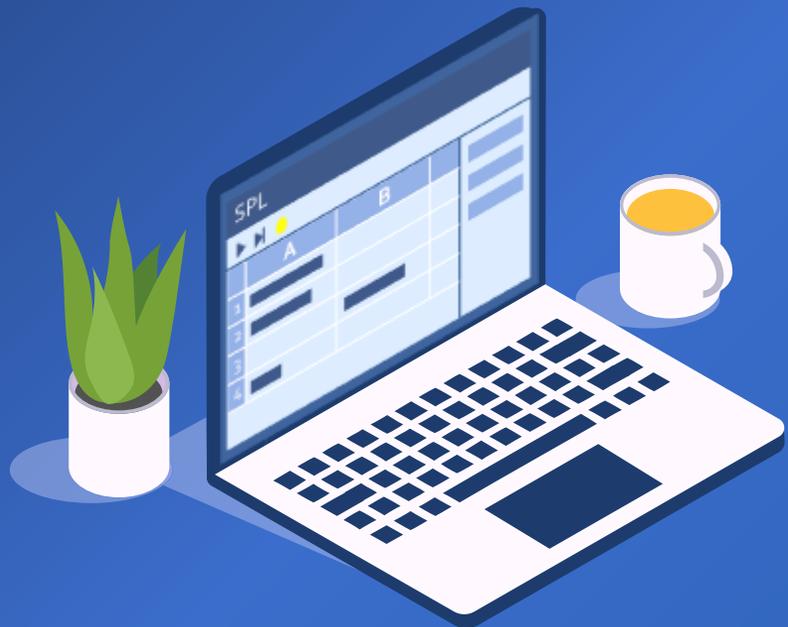
Concatenate Details as a table sequence:

| | result | Details |
|---|--------|---------------------------|
| 1 | true | [[301500,2023-07-21,69.6, |
| 2 | true | |
| 3 | true | |
| 4 | false | |
| 5 | true | |
| 6 | true | |
| 7 | true | |
| 8 | true | |

Final result set:

| | code | date | open | close |
|----|--------|------------|-------|-------|
| 1 | 301500 | 2023-07-21 | 69.6 | 73.86 |
| 2 | 301500 | 2023-07-24 | 67.25 | 74.58 |
| 3 | 301500 | 2023-07-25 | 88.47 | 81.11 |
| 4 | 301500 | 2023-07-26 | 78.58 | 82.28 |
| 5 | 301500 | 2023-07-27 | 88.57 | 89.42 |
| 6 | 320688 | 2023-06-29 | 17.65 | 16.76 |
| 7 | 320688 | 2023-06-30 | 16.49 | 17.58 |
| 8 | 320688 | 2023-07-03 | 19.2 | 19.09 |
| 9 | 320688 | 2023-07-04 | 19.03 | 19.76 |
| 10 | 320688 | 2023-07-05 | 20.32 | 21.51 |
| 11 | 320688 | 2023-07-21 | 23.65 | 24.92 |
| 12 | 320688 | 2023-07-24 | 24.27 | 26.68 |
| 13 | 320688 | 2023-07-25 | 25.21 | 27.51 |
| 14 | 320688 | 2023-07-26 | 31.57 | 29.66 |

CONTENTS



**Compute LRR and YOY
based on car sales table**

4.2 Compute LRR and YOY based on car sales table



Create a grouped report according to car sales table to display each model's LRR and YOY in every month in three years. Below is data in [Sale.4.2.xlsx](#):

| | ID | Model | Date |
|----|----|---------|------------|
| 1 | 1 | Golf | 2010-01-01 |
| 2 | 2 | CC | 2010-01-01 |
| 3 | 3 | CC | 2010-01-02 |
| 4 | 4 | Touareg | 2010-01-02 |
| 5 | 5 | Touareg | 2010-01-02 |
| 6 | 6 | CC | 2010-01-02 |
| 7 | 7 | Jetta | 2010-01-02 |
| 8 | 8 | CC | 2010-01-02 |
| 9 | 9 | Golf | 2010-01-02 |
| 10 | 10 | Golf | 2010-01-04 |
| 11 | 11 | Jetta | 2010-01-04 |
| 12 | 12 | Touareg | 2010-01-04 |
| 13 | 13 | Touareg | 2010-01-04 |
| 14 | 14 | CC | 2010-01-04 |

4.2 Compute LRR and YOY based on car sales table



Group Sale table by model, month and year:

Group dialog box configuration:

| Index | Field expression |
|-------|-------------------|
| 1 | Model |
| 2 | month(Date):Month |
| 3 | year(Date):Year |

Compute section:

| Compute |
|---------|
| Date |
| ID |
| Model |

Expression: month(Date):Month

Set aggregate expression:

| Index | Aggregate expression | Alias |
|-------|----------------------|----------|
| 1 | count(ID) | SalesVol |

4.2 Compute LRR and YOY based on car sales table



Grouping result set:

| | Model | Month | Year | SalesVol |
|----|-------|-------|------|----------|
| 1 | CC | 1 | 2010 | 26 |
| 2 | CC | 1 | 2011 | 26 |
| 3 | CC | 1 | 2012 | 31 |
| 4 | CC | 2 | 2010 | 22 |
| 5 | CC | 2 | 2011 | 29 |
| 6 | CC | 2 | 2012 | 39 |
| 7 | CC | 3 | 2010 | 29 |
| 8 | CC | 3 | 2011 | 27 |
| 9 | CC | 3 | 2012 | 31 |
| 10 | CC | 4 | 2010 | 33 |
| 11 | CC | 4 | 2011 | 33 |
| 12 | CC | 4 | 2012 | 33 |
| 13 | CC | 5 | 2010 | 27 |
| 14 | CC | 5 | 2011 | 33 |
| 15 | CC | 5 | 2012 | 28 |

Compute YOY:

Computed column dialog box details:

- Name: SalesMY
- Source table: SalesMY
- Computed column table:

| Index | Expression | Alias |
|-------|---|-------|
| 1 | if(Model==Model[-1] && Month==Month[-1],SalesVol/SalesVol[-1],null) | YOY |

Expression:if(Model==Model[-1] && Month==Month[-1],SalesVol/SalesVol[-1],null)

4.2 Compute LRR and YOY based on car sales table



Result set:

| | Model | Month | Year | SalesVol | YOY |
|----|-------|-------|------|----------|--------------------|
| 1 | CC | 1 | 2010 | 26 | (null) |
| 2 | CC | 1 | 2011 | 26 | 1.0 |
| 3 | CC | 1 | 2012 | 31 | 1.1923076923076923 |
| 4 | CC | 2 | 2010 | 22 | (null) |
| 5 | CC | 2 | 2011 | 29 | 1.3181818181818181 |
| 6 | CC | 2 | 2012 | 39 | 1.3448275862068966 |
| 7 | CC | 3 | 2010 | 29 | (null) |
| 8 | CC | 3 | 2011 | 27 | 0.9310344827586207 |
| 9 | CC | 3 | 2012 | 31 | 1.1481481481481481 |
| 10 | CC | 4 | 2010 | 33 | (null) |
| 11 | CC | 4 | 2011 | 33 | 1.0 |
| 12 | CC | 4 | 2012 | 40 | 1.2121212121212122 |
| 13 | CC | 5 | 2010 | 21 | (null) |
| 14 | CC | 5 | 2011 | 33 | 1.5714285714285714 |
| 15 | CC | 5 | 2012 | 28 | 0.8484848484848485 |
| 16 | CC | 6 | 2010 | 20 | (null) |

4.2 Compute LRR and YOY based on car sales table



First, select Year field and shift it leftward to the front of Month field:

| | Model | Month | Year | SalesVol | YOY |
|----|-------|-------|------|----------|--------------------|
| 1 | CC | 1 | 2010 | 26 | (null) |
| 2 | CC | 1 | 2011 | 26 | 1.0 |
| 3 | CC | 1 | 2012 | 31 | 1.1923076923076923 |
| 4 | CC | 2 | 2010 | 22 | (null) |
| 5 | CC | 2 | 2011 | 29 | 1.3181818181818181 |
| 6 | CC | 2 | 2012 | 39 | 1.3448275862068966 |
| 7 | CC | 3 | 2010 | 29 | (null) |
| 8 | CC | 3 | 2011 | 27 | 0.9310344827586207 |
| 9 | CC | 3 | 2012 | 31 | 1.1481481481481481 |
| 10 | CC | 4 | 2010 | 33 | (null) |
| 11 | CC | 4 | 2011 | 33 | 1.0 |
| 12 | CC | 4 | 2012 | 40 | 1.2121212121212122 |
| 13 | CC | 5 | 2010 | 21 | (null) |
| 14 | CC | 5 | 2011 | 33 | 1.5714285714285714 |

4.2 Compute LRR and YOY based on car sales table



Then sort rows by Model, Year and Month, and change table name to SalesYM:

The screenshot shows a data tool interface with a table and a 'Sort' dialog box. The table has columns: Model, Year, Month, SalesVol, and YOY. The 'Sort' dialog box is open, showing the following configuration:

- Name: SalesYM (highlighted with a red box)
- Source table: SalesMY
- Locale: English
- Sorting field: Model, Year, Month (all with checkboxes checked for ascending sort)
- Source fields (Double click to select): Model, Month, SalesVol, YOY, Year
- Options: Parallel computation (unchecked), Place null values at the end (unchecked)

| | Model | Year | Month | SalesVol | YOY |
|---|-------|------|-------|----------|--------|
| 1 | CC | 2010 | 1 | 26 | (null) |

4.2 Compute LRR and YOY based on car sales table



Sorting result set:

| | Model | Year | Month | SalesVol | YOY |
|----|-------|------|-------|----------|--------------------|
| 1 | CC | 2010 | 1 | 26 | (null) |
| 2 | CC | 2010 | 2 | 22 | (null) |
| 3 | CC | 2010 | 3 | 29 | (null) |
| 4 | CC | 2010 | 4 | 33 | (null) |
| 5 | CC | 2010 | 5 | 21 | (null) |
| 6 | CC | 2010 | 6 | 20 | (null) |
| 7 | CC | 2010 | 7 | 21 | (null) |
| 8 | CC | 2010 | 8 | 36 | (null) |
| 9 | CC | 2010 | 9 | 21 | (null) |
| 10 | CC | 2010 | 10 | 43 | (null) |
| 11 | CC | 2010 | 11 | 23 | (null) |
| 12 | CC | 2010 | 12 | 15 | (null) |
| 13 | CC | 2011 | 1 | 26 | |
| 14 | CC | 2011 | 2 | 29 | 1.3181818181818181 |
| 15 | CC | 2011 | 3 | 27 | 0.9310344827586207 |

Compute LRR:

Computed column dialog box details:

- Name: SalesYM
- Source table: SalesYM
- Computed column table:

| Index | Expression | Alias |
|-------|---|-------|
| 1 | if(Model==Model[-1],SalesVol/SalesVol[-1],null) | LRR |

Expression: if(Model==Model[-1],SalesVol/SalesVol[-1],null)

◆ 4.2 Compute LRR and YOY based on car sales table



Final result:

| | Model | Year | Month | SalesVol | YOY | LRR |
|----|-------|------|-------|----------|--------------------|--------------------|
| 1 | CC | 2010 | 1 | 26 | (null) | (null) |
| 2 | CC | 2010 | 2 | 22 | (null) | 0.8461538461538461 |
| 3 | CC | 2010 | 3 | 29 | (null) | 1.3181818181818181 |
| 4 | CC | 2010 | 4 | 33 | (null) | 1.1379310344827587 |
| 5 | CC | 2010 | 5 | 21 | (null) | 0.6363636363636364 |
| 6 | CC | 2010 | 6 | 20 | (null) | 0.9523809523809523 |
| 7 | CC | 2010 | 7 | 21 | (null) | 1.05 |
| 8 | CC | 2010 | 8 | 36 | (null) | 1.7142857142857142 |
| 9 | CC | 2010 | 9 | 21 | (null) | 0.5833333333333334 |
| 10 | CC | 2010 | 10 | 43 | (null) | 2.0476190476190474 |
| 11 | CC | 2010 | 11 | 23 | (null) | 0.5348837209302325 |
| 12 | CC | 2010 | 12 | 15 | (null) | 0.6521739130434783 |
| 13 | CC | 2011 | 1 | 26 | 1.0 | 1.7333333333333334 |
| 14 | CC | 2011 | 2 | 29 | 1.3181818181818181 | 1.1153846153846154 |
| 15 | CC | 2011 | 3 | 27 | 0.9310344827586207 | 0.9310344827586207 |
| 16 | CC | 2011 | 4 | 33 | 1.0 | 1.2222222222222223 |
| 17 | CC | 2011 | 5 | 33 | 1.5714285714285714 | 1.0 |
| 18 | CC | 2011 | 6 | 25 | 1.25 | 0.7575757575757576 |

CONTENTS



Associate worksheets and compute salaries

4.3 Associate worksheets and compute salaries



Task: Combine performance table and absent table with standard table, and compute the actual salary. Below are the three worksheets in **Employee.4.3.xlsx**:

| | EID | Name | StandardWages |
|----|-----|----------------|---------------|
| 1 | 1 | Rebecca Moore | 2000 |
| 2 | 2 | Ashley wilson | 2200 |
| 3 | 3 | Rachel Johnson | 1800 |
| 4 | 4 | Emily Smith | 1200 |
| 5 | 5 | Ashley scott | 2000 |
| 6 | 6 | Matthew Jones | 1600 |
| 7 | 7 | Alexis Smith | 1300 |
| 8 | 8 | Megan wilson | 3000 |
| 9 | 9 | Victoria Green | 2300 |
| 10 | 10 | Ryan Jackson | 2600 |
| 11 | 11 | Jacob Moore | 1250 |
| 12 | 12 | Jessica Davis | 2000 |
| 13 | 13 | Daniel Davis | 2800 |
| 14 | 14 | Alyssa Wilson | 1800 |

| | EID | Absence |
|---|-----|---------|
| 1 | 2 | 8 |
| 2 | 11 | 15 |
| 3 | 13 | 20 |
| 4 | 17 | 4 |
| 5 | 19 | 12 |

| | EID | Evaluation | Bonus |
|---|-----|------------|-------|
| 1 | 2 | 0 | 800 |
| 2 | 3 | 0 | 500 |
| 3 | 5 | 0 | 200 |
| 4 | 7 | 0.2 | 1500 |
| 5 | 11 | -0.15 | 0 |
| 6 | 13 | 0 | 300 |
| 7 | 17 | 0.05 | 500 |
| 8 | 19 | 0.3 | 800 |

4.3 Associate worksheets and compute salaries



Perform left join to associate absent table and performance table with standard table through join field EID:

| | EID | Name | StandardWages |
|---|-----|---------------|---------------|
| 1 | 1 | Rebecca Moore | 2000 |

Name: **All** Source table: standard

Join type: Inner join **Left join**

| Index | Target table | Select |
|-------|--------------|-------------------------------------|
| 1 | absent | <input checked="" type="checkbox"/> |
| 2 | performance | <input checked="" type="checkbox"/> |

Set join field:

| Index | standard | absent | performance |
|-------|----------|--------|-------------|
| 1 | EID | EID | EID |

Check field to be selected:

| Index | Table | Field | Select | Alias |
|-------|-------------|---------------|-------------------------------------|-------|
| 1 | standard | EID | <input checked="" type="checkbox"/> | |
| 2 | standard | Name | <input checked="" type="checkbox"/> | |
| 3 | standard | StandardWages | <input checked="" type="checkbox"/> | |
| 4 | absent | EID | <input type="checkbox"/> | |
| 5 | absent | Absence | <input checked="" type="checkbox"/> | |
| 6 | performance | EID | <input type="checkbox"/> | |
| 7 | performance | Evaluation | <input checked="" type="checkbox"/> | |
| 8 | performance | Bonus | <input checked="" type="checkbox"/> | |

4.3 Associate worksheets and compute salaries



Association result set:

| | EID | Name | StandardWage | Absence | Evaluation | Bonus |
|----|-----|-----------------|--------------|---------|------------|--------|
| 1 | 1 | Rebecca Moore | 2000 | (null) | (null) | (null) |
| 2 | 2 | Ashley wilson | 2200 | 8 | 0 | 800 |
| 3 | 3 | Rachel Johnson | 1800 | (null) | 0 | 500 |
| 4 | 4 | Emily Smith | 1200 | (null) | (null) | (null) |
| 5 | 5 | Ashley scott | 2000 | (null) | 0 | 200 |
| 6 | 6 | Matthew Jones | 1600 | (null) | (null) | (null) |
| 7 | 7 | Alexis Smith | 1300 | (null) | 0.2 | 1500 |
| 8 | 8 | Megan wilson | 3000 | (null) | (null) | (null) |
| 9 | 9 | Victoria Green | 2300 | (null) | (null) | (null) |
| 10 | 10 | Ryan Jackson | 2600 | (null) | (null) | (null) |
| 11 | 11 | Jacob Moore | 1250 | 15 | -0.15 | 0 |
| 12 | 12 | Jessica Davis | 2000 | (null) | (null) | (null) |
| 13 | 13 | Daniel Davis | 2800 | 20 | 0 | 300 |
| 14 | 14 | Alyssa Wilson | 1800 | (null) | (null) | (null) |
| 15 | 15 | Alexis Sullivan | 2400 | (null) | (null) | (null) |

4.3 Associate worksheets and compute salaries



Add a computed column to ALL table, and compute salaries, whose field alias is Salary:

The screenshot shows a data management tool interface. At the top, there are tabs for 'standard', 'absent', 'performance', and 'All'. Below the tabs is a toolbar with various icons, including a filter icon (ABC 123) which is circled in red. The main area displays a table with the following columns: EID, Name, StandardWages, Absence, Evaluation, and Bonus. The first row contains the data: 1, Rebecca Moore, 2000, (null), (null), (null). A 'Computed column' dialog box is open, showing the following fields:

- Name: All
- Source table: All
- Options: (empty)
- Computed column: (empty)
- Source fields (Double click to select): Absence, Bonus, EID, Evaluation, Name, StandardWages

The 'Computed column' dialog box has a table with the following columns: Index, Expression, and Alias. The first row is highlighted in yellow and contains the following data:

| Index | Expression | Alias |
|-------|------------------------------|--------|
| 1 | StandardWages*(1+Evaluation) | Salary |

A red box highlights the 'Expression' and 'Alias' columns of this table. A red arrow points from the 'Expression' cell to a text box below the dialog box containing the following text:

Expression: StandardWages*(1+Evaluation-Absence/40)+Bonus

4.3 Associate worksheets and compute salaries



Compute Salary field values:

| | EID | Name | StandardWages | Absence | Evaluation | Bonus | Salary |
|----|-----|-----------------|---------------|---------|------------|--------|--------|
| 1 | 1 | Rebecca Moore | 2000 | (null) | (null) | (null) | 2000 |
| 2 | 2 | Ashley wilson | 2200 | 8 | 0 | 800 | 2560.0 |
| 3 | 3 | Rachel Johnson | 1800 | (null) | 0 | 500 | 2300 |
| 4 | 4 | Emily Smith | 1200 | (null) | (null) | (null) | 1200 |
| 5 | 5 | Ashley scott | 2000 | (null) | 0 | 200 | 2200 |
| 6 | 6 | Matthew Jones | 1600 | (null) | (null) | (null) | 1600 |
| 7 | 7 | Alexis Smith | 1300 | (null) | 0.2 | 1500 | 3060.0 |
| 8 | 8 | Megan wilson | 3000 | (null) | (null) | (null) | 3000 |
| 9 | 9 | Victoria Green | 2300 | (null) | (null) | (null) | 2300 |
| 10 | 10 | Ryan Jackson | 2600 | (null) | (null) | (null) | 2600 |
| 11 | 11 | Jacob Moore | 1250 | 15 | -0.15 | 0 | 593.75 |
| 12 | 12 | Jessica Davis | 2000 | (null) | (null) | (null) | 2000 |
| 13 | 13 | Daniel Davis | 2800 | 20 | 0 | 300 | 1700.0 |
| 14 | 14 | Alyssa Wilson | 1800 | (null) | (null) | (null) | 1800 |
| 15 | 15 | Alexis Sullivan | 2400 | (null) | (null) | (null) | 2400 |

CONTENTS



**Find salespeople who rank
top5 every month**

✦ 4.4 Find salespeople who rank top5 every month



According to the daily trading records, find star salespeople who make it to top 5 every month in terms of sales volume. Open data table **Trade.4.4.xlsx** as follows:

| | ID | Date | Sale | Value |
|----|----|------------|---------|-------|
| 1 | 1 | 2012-01-01 | Baker | 3233 |
| 2 | 2 | 2012-01-01 | Marry | 6750 |
| 3 | 3 | 2012-01-01 | Jenny | 2998 |
| 4 | 4 | 2012-01-01 | Steven | 8059 |
| 5 | 5 | 2012-01-01 | Bill | 6933 |
| 6 | 6 | 2012-01-01 | Tom | 4828 |
| 7 | 7 | 2012-01-01 | Joancy | 9232 |
| 8 | 8 | 2012-01-01 | Larren | 2733 |
| 9 | 9 | 2012-01-01 | Harry | 5197 |
| 10 | 10 | 2012-01-01 | Johnson | 2878 |
| 11 | 11 | 2012-01-02 | Harry | 8940 |
| 12 | 12 | 2012-01-02 | Joancy | 5707 |
| 13 | 13 | 2012-01-03 | Marry | 2687 |
| 14 | 14 | 2012-01-03 | Jenny | 7503 |
| 15 | 15 | 2012-01-03 | Joancy | 166 |

4.4 Find salespeople who rank top5 every month



Group Trade table by Month and Sales, and compute each salesperson's sales volume in each month:

Trade

Group

Name: **groupMonth** Source table: Trade

Options: [Buttons]

Aggregates: **sum** Source fields (Double click to select)

| Index | Field expression |
|-------|--------------------------|
| 1 | month(Date):Month |
| 2 | Sale |

表达式: **month(Date):Month**

Keep the original order
 Return the first row of each group
 Group when the key changes
 Parallel computation

Discard groups with null key
 Group when the key is true
 Retain details

Set aggregation expression:

Aggregates: **sum**

| Index | Aggregate expression | Alias |
|-------|----------------------|------------|
| 1 | sum(Value) | GrandTotal |

4.4 Find salespeople who rank top5 every month



Grouping result set:

| | Month | Sale | GrandTotal |
|----|-------|---------|------------|
| 1 | 1 | Baker | 82947 |
| 2 | 1 | Bill | 106036 |
| 3 | 1 | Dow | 100529 |
| 4 | 1 | Harry | 79994 |
| 5 | 1 | Jenny | 108439 |
| 6 | 1 | Joancy | 61735 |
| 7 | 1 | Johnson | 60501 |
| 8 | 1 | Larren | 95658 |
| 9 | 1 | Marry | 60315 |
| 10 | 1 | Steven | 95644 |
| 11 | 1 | Tom | 80613 |
| 12 | 2 | Baker | 89837 |
| 13 | 2 | Bill | 99535 |
| 14 | 2 | Dow | 98712 |
| 15 | 2 | Harry | 59620 |

4.4 Find salespeople who rank top5 every month



Then group groupMonth table by Month, and name result set MonthTop5:

The screenshot shows a data tool interface with a table and a 'Group' dialog box. The table has columns: Month, Sale, GrandTotal. The 'Group' dialog box has the following fields:

- Name: MonthTop5
- Source table: groupMonth
- Aggregates: sum
- Grouping field: Month
- Source fields (Double click to select): GrandTotal, Month, Sale

At the bottom of the dialog, there are several checkboxes:

- Keep the original order
- Return the first row of each group
- Group when the key changes
- Parallel computation
- Grouping key is sequence number
- Discard groups with null key
- Group when the key is true
- Retain details

Set aggregation expression:

The screenshot shows the 'Aggregates' tab of the dialog box. It has a table with the following columns: Index, Aggregate expression, Alias.

| Index | Aggregate expression | Alias |
|-------|----------------------|-------|
| 1 | top(-5;GrandTotal) | Top5 |

Expression: top(-5;GrandTotal)

4.4 Find salespeople who rank top5 every month



Grouping result set:

| | Month | Top5 |
|----|-------|---|
| 1 | 1 | [[1,Jenny],[1,Bill],[1,Dow], ...] |
| 2 | 2 | [[2,Johnson],[2,Bill],[2,Dow], ...] |
| 3 | 3 | [[3,Tom],[3,Baker],[3,Bill], ...] |
| 4 | 4 | [[4,Bill],[4,Johnson],[4,Larren], ...] |
| 5 | 5 | [[5,Tom],[5,Dow],[5,Bill], ...] |
| 6 | 6 | [[6,Johnson],[6,Larren],[6,Mary], ...] |
| 7 | 7 | [[7,Dow],[7,Bill],[7,Baker], ...] |
| 8 | 8 | [[8,Larren],[8,Bill],[8,Tom], ...] |
| 9 | 9 | [[9,Johnson],[9,Larren],[9,Harry], ...] |
| 10 | 10 | [[10,Steven],[10,Tom],[10,Dow], ...] |
| 11 | 11 | [[11,Bill],[11,Baker],[11,Jenny], ...] |
| 12 | 12 | [[12,Tom],[12,Marry],[12,Bill], ...] |

Top5 salespeople
in each month

4.4 Find salespeople who rank top5 every month



Compute intersection of all top5s:

| | Month | Top5 |
|----|-------|---|
| 1 | 1 | [[1,Jenny],[1,Bill],[1,Dow], ...] |
| 2 | 2 | [[2,Johnson],[2,Bill],[2,Dow], ...] |
| 3 | 3 | [[3,Tom],[3,Baker],[3,Bill], ...] |
| 4 | 4 | [[4,Bill],[4,Johnson],[4,Larren], ...] |
| 5 | 5 | [[5,Tom],[5,Dow],[5,Bill], ...] |
| 6 | 6 | [[6,Johnson],[6,Larren],[6,Harry], ...] |
| 7 | 7 | [[7,Dow],[7,Bill],[7,Baker], ...] |
| 8 | 8 | [[8,Larren],[8,Bill],[8,Tom], ...] |
| 9 | 9 | [[9,Johnson],[9,Larren],[9,Harry], ...] |
| 10 | 10 | [[10,Steven],[10,Tom],[10,Dow], ...] |
| 11 | 11 | [[11,Bill],[11,Baker],[11,Jenny], ...] |
| 12 | 12 | [[12,Tom],[12,Mary],[12,Bill], ...] |

Expression:
`MonthTop5.(Top5.(Sale)).isect()`

Enter the expression at command line and execute it

The best salesperson:

| Index | Member |
|-------|--------|
| 1 | Bill |

SPL WIN Course

Chapter 5

Editing result set

SPL WIN



✦ Editing result set



A result set meeting the following conditions is view-only and cannot be edited:

An unnamed table sequence returned from executing an expression at the command line

| Index | Date | Sale | Value |
|-------|---------------|---------|-------|
| 1 | 2012-01-01 | Baker | 3233 |
| 2 | | | 6750 |
| 3 | | | 2998 |
| 4 | | | 59 |
| 5 | | | 8 |
| 6 | | | 9232 |
| 8 | 8 2012-0 | | 2733 |
| 9 | 9 2012-0 | Harry | 5197 |
| 10 | 10 2012-0 | Johnson | 2878 |
| 11 | 11 2012-0 | Harry | 8940 |
| 12 | 12 2012-01-02 | Joancy | 5707 |

Edit command Enable prompt

T("Trade.4.4.xlsx")

A sequence result cannot be edited even it is named

| Index | Member |
|-------|---------|
| 1 | Baker |
| 2 | Marry |
| 3 | Jenny |
| 4 | Steven |
| 5 | Bill |
| 6 | Tom |
| 7 | Joancy |
| 8 | Larren |
| 9 | Harry |
| 10 | Johnson |
| 11 | Harry |
| 12 | Joancy |

Edit command Enable prompt

Trade.(Sale)

✦ Editing result set



A result set meeting the following conditions is view-only and cannot be edited :

result Trade group

| | Month | Details |
|----|-------|---------------------------|
| 1 | | [[1,2012-01-01,Baker, |
| 2 | | [[186,2012-02-01,Steven, |
| 3 | | [[356,2012-03-01,Larren, |
| 4 | | [[532,2012-04-01,Baker, |
| 5 | | [[737,2012-05-01,Jenny, |
| 6 | | [[890,2012-06-01,Johnson, |
| 7 | | [[1087,2012-07-01,Harry, |
| 8 | | [[1311,2012-08-01,Harry, |
| 9 | | [[1506,2012-09-01,Jenny, |
| 10 | | [[1683,2012-10-01,Larren, |
| 11 | | [[1846,2012-11-01,Larren, |
| 12 | | [[2054,2012-12-01,Larren, |

A field whose values are sequences or table sequences cannot be edited

Trade

| | ID | Date | | |
|----|----|------------|---------|------|
| 1 | 1 | 2012-01-01 | E | |
| 2 | 2 | 2012-01-01 | Ma | |
| 3 | 3 | 2012-01-01 | Jenny | |
| 4 | 4 | 2012-01-01 | Steven | |
| 5 | 5 | 2012-01-01 | Bill | 6933 |
| 6 | 6 | 2012-01-01 | Tom | 4828 |
| 7 | 7 | 2012-01-01 | Joancy | 9232 |
| 8 | 8 | 2012-01-01 | Larren | 2733 |
| 9 | 9 | 2012-01-01 | Harry | 5197 |
| 10 | 10 | 2012-01-01 | Johnson | 2878 |
| 11 | 11 | 2012-01-02 | Harry | 8940 |
| 12 | 12 | 2012-01-02 | Jenny | 5707 |

When a data table is stored as a cursor, all its fields cannot be edited

CONTENTS



5.1 Data rows

5.2 Data columns

5.3 Cell value expression

5.4 Save and export

CONTENTS



Data rows

✦ 5.1 Data rows



We can move data rows, or add or delete one:

| | ID | Date | Sale | Value |
|----|----|------------|---------|-------|
| 1 | 1 | 2012-01-01 | Baker | 3233 |
| 2 | 2 | 2012-01-01 | Marry | 6750 |
| 3 | 3 | 2012-01-01 | Jenny | 2998 |
| 4 | 4 | 2012-01-01 | Steven | 8059 |
| 5 | 5 | 2012-01-01 | Bill | 6933 |
| 6 | 6 | 2012-01-01 | Tom | 4828 |
| 7 | 7 | 2012-01-01 | Joancy | 9232 |
| 8 | 8 | 2012-01-01 | Larren | 2733 |
| 9 | 9 | 2012-01-01 | Harry | 5197 |
| 10 | 10 | 2012-01-01 | Johnson | 287 |
| 11 | 11 | 2012-01-02 | Harry | 8940 |
| 12 | 12 | 2012-01-02 | Joancy | 5707 |

Click a row's head cell to select the row for performing operations on it

Only when a row is selected will the corresponding toolbar icons are enabled
Note: The rule is invalid when result is a cursor

CONTENTS



Data columns

✦ 5.2 Data columns



We can move data columns, or add or delete one:

The screenshot shows a data table with columns: ID, Date, Sale, and Value. The 'Sale' column header is circled in red. A toolbar above the table contains icons for filtering, sorting, and column operations. A blue box highlights the column operation icons: left arrow, right arrow, add column, delete column, and close. Two yellow callouts provide instructions: one points to the 'Sale' header, and the other points to the column operation icons.

| | ID | Date | Sale | Value |
|----|----|------------|---------|-------|
| 1 | 1 | 2012-01-01 | Peter | 3233 |
| 2 | 2 | 2012-01-01 | Marry | 6750 |
| 3 | 3 | 2012-01-01 | Jenny | 2998 |
| 4 | 4 | 2012-01-01 | Steven | 8059 |
| 5 | 5 | 2012-01-01 | Bill | 6933 |
| 6 | 6 | 2012-01-01 | Tom | 4828 |
| 7 | 7 | 2012-01-01 | Joancy | 9232 |
| 8 | 8 | 2012-01-01 | Larren | 2733 |
| 9 | 9 | 2012-01-01 | Harry | 5197 |
| 10 | 10 | 2012-01-01 | Johnson | 2878 |
| 11 | 11 | 2012-01-02 | Harry | 8940 |
| 12 | 12 | 2012-01-02 | Jenny | 5707 |

Click a column's head cell to select the column for performing operations on it

Only when a column is selected will the corresponding toolbar icons are enabled
Note: The rule is invalid when result is a cursor

CONTENTS



Cell value expression

✦ 5.3 Cell value expression



We can directly assign value to a cell, or compute its value through an expression beginning with the equals sign (the latter becomes disabled when result is a cursor):

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|--------|-------|-------------|
| 1 | 1 | 2012-01-01 | Baker | 3233 | =Value*0.05 |
| 2 | 2 | 2012-01-01 | Marry | 6750 | (null) |
| 3 | 3 | 2012-01-01 | Jenny | 2998 | (null) |
| 4 | 4 | 2012-01-01 | Steven | 8059 | (null) |
| 5 | 5 | 2012-01-01 | Bill | 6900 | (null) |
| 6 | 6 | 2012-01-01 | Tom | 4828 | (null) |
| 7 | 7 | 2012-01-01 | Joancy | 9232 | (null) |
| 8 | 8 | 2012-01-01 | | | (null) |
| 9 | 9 | 2012-01-01 | | | (null) |
| 10 | 10 | 2012-01-01 | | | (null) |
| 11 | 11 | 2012-01-01 | | | (null) |

The equals sign indicates an expression that computes only for the current row

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|--------|-------|--------|
| 1 | 1 | 2012-01-01 | Baker | 3233 | 161.65 |
| 2 | 2 | 2012-01-01 | Marry | 6750 | (null) |
| 3 | 3 | 2012-01-01 | Jenny | 2998 | (null) |
| 4 | 4 | 2012-01-01 | Steven | 8059 | (null) |
| 5 | 5 | 2012-01-01 | Bill | 6900 | (null) |
| 6 | 6 | 2012-01-01 | Tom | 4828 | (null) |
| 7 | 7 | 2012-01-01 | Joancy | 9232 | (null) |
| 8 | 8 | 2012-01-01 | | | (null) |
| 9 | 9 | 2012-01-01 | | | (null) |
| 10 | 10 | 2012-01-01 | | | (null) |
| 11 | 11 | 2012-01-01 | | | (null) |

Compute bonus according to the sales volume

✦ 5.3 Cell value expression



We can also use an expression beginning with the double equals sign to compute (this is disabled when result is a cursor):

The screenshot shows a table with columns ID, Date, Sale, Value, and Bonus. The Bonus column is selected, and the formula `==Value*0.05` is being entered into the first cell. A yellow callout bubble points to the formula.

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|--------|-------|---------------------------|
| 1 | 1 | 2012-01-01 | Baker | 3233 | <code>==Value*0.05</code> |
| 2 | 2 | 2012-01-01 | Marry | 6750 | (null) |
| 3 | 3 | 2012-01-01 | Jenny | 2998 | (null) |
| 4 | 4 | 2012-01-01 | Steven | 8059 | (null) |
| 5 | 5 | 2012-01-01 | Bill | 6933 | (null) |
| 6 | 6 | 2012-01-01 | Tom | 828 | (null) |
| 7 | 7 | 2012-01-01 | Joancy | 9232 | (null) |
| 8 | | | | 2733 | (null) |
| 9 | | | | | (null) |
| 10 | | | | | (null) |
| 11 | | | | | (null) |

The double equals sign indicates an expression that computes value for the whole column

The screenshot shows the same table as the previous one, but now the Bonus column contains calculated values. A blue box highlights the Bonus column, and a yellow callout bubble points to it.

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|--------|-------|---------------|
| 1 | 1 | 2012-01-01 | Baker | 3233 | 161.65 |
| 2 | 2 | 2012-01-01 | Marry | 6750 | 337.5 |
| 3 | 3 | 2012-01-01 | Jenny | 2998 | 149.9 |
| 4 | 4 | 2012-01-01 | Steven | 8059 | 402.950000000 |
| 5 | 5 | 2012-01-01 | Bill | 6933 | 346.650000000 |
| 6 | 6 | 2012-01-01 | Tom | 828 | 241.4 |
| 7 | 7 | 2012-01-01 | Joancy | 9232 | 461.6 |
| 8 | | | | 2733 | 136.65 |
| 9 | | | | 5197 | 259.85 |
| 10 | | | | 2878 | 143.9 |
| 11 | | | | 8940 | 447.0 |

Compute bonuses for all salespeople according sales volumes

CONTENTS



Save and export

✦ 5.4 Save and export



A table sequence or a cursor obtained from the computation can be directly saved as a text file:

The screenshot shows a software window titled 'Trade' containing a table with the following data:

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|---------|-------|--------|
| 1 | 1 | 2012-01-01 | Baker | 3233 | 161.65 |
| 2 | 2 | 2012-01-01 | Marry | | |
| 3 | 3 | 2012-01-01 | Jenny | | |
| 4 | 4 | 2012-01-01 | Steven | | |
| 5 | 5 | 2012-01-01 | Bill | | |
| 6 | 6 | 2012-01-01 | Tom | | |
| 7 | 7 | 2012-01-01 | Joancy | | |
| 8 | 8 | 2012-01-01 | Larren | | |
| 9 | 9 | 2012-01-01 | Harry | | |
| 10 | 10 | 2012-01-01 | Johnson | | |
| 11 | 11 | 2012-01-02 | Harry | | |

A 'Save as' dialog box is open, showing the 'Look In' field set to 'tutorial' and a folder named 'data'. The 'Files of Type' is set to '*.txt'. A yellow callout bubble points to the 'Save' icon in the toolbar, with the text: 'Click the Save icon to save a table sequence/cursor as a text file'.

◆ 5.4 Save and export



They can also be exported as a file of another format:

Trade

| | ID | Date | Sale | Value | Bonus |
|----|----|------------|--------|-------|-------|
| 1 | 1 | 2012-01-01 | Baker | | |
| 2 | 2 | 2012-01-01 | Marry | | |
| 3 | 3 | 2012-01-01 | Jenny | | |
| 4 | 4 | 2012-01-01 | Steven | | |
| 5 | 5 | 2012-01-01 | Bill | | |
| 6 | 6 | 2012-01-01 | Tom | | |
| 7 | 7 | 2012-01-01 | Joancy | | |
| 8 | 8 | 2012-01-01 | Larren | | |
| 9 | 9 | 2012-01-01 | Harry | | |
| 10 | | | | | |
| 11 | | | | | |

Export

Name: export Source table: Trade

File name: []

Files of type: txt (dropdown menu open showing: txt, cbx, btx, txt, csv, xlsx)

Options: t

Table name: []

| Index | Field | Select | Key |
|-------|-------|-------------------------------------|--------------------------|
| | Date | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| | Sale | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| | Value | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| | Bonus | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

Export column headers Append

Write to binary file f by segment Use windows-style line break

Enclose field values and header... Use double quotation marks as escap...

Click **Export** icon to export data

There are more file format choices in the Export panel