The Finer Things In Alteryx

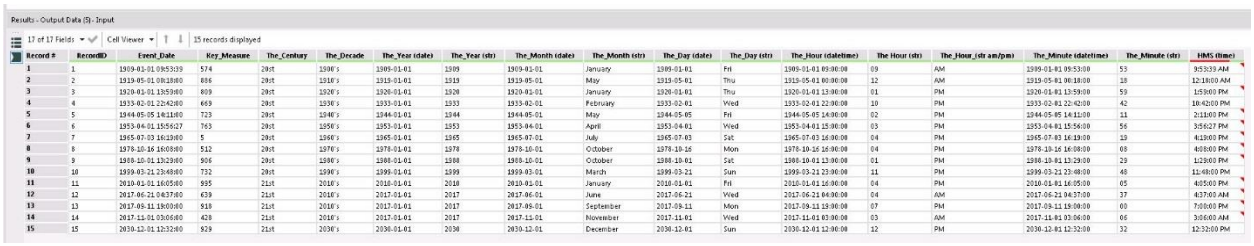
Ken Black 7/2/18

Controlling Level Of Detail by Parsing Dates

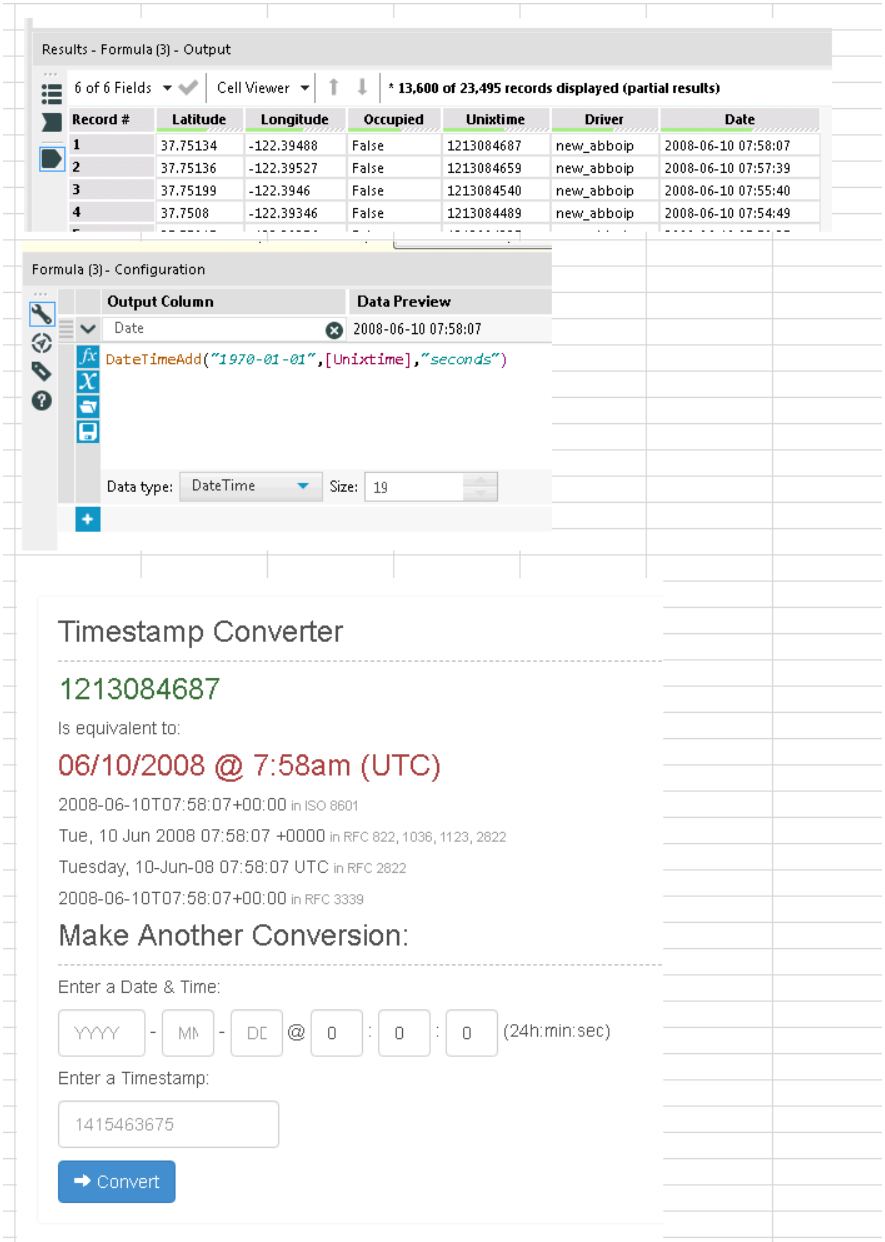
The_Century - tostring(tonumber((left(DateTimeFormat([Event_Date],'%Y'),2))) + 1) + "st"

The_Decade - left(DateTimeFormat([Event_Date],'%Y'),3) + "0's"


The_Day (date)
1909-01-01
// This stips off the minutes/seconds
DateTimeFormat([Event_Date],'%Y-%m-%d')
Data type: Date
Size: 10
The_Day (str)
Fri
// This returns the Day
// a = Fri
// A = Friday
// d = 06
//
DateTimeFormat([Event_Date],'%a')
Data type: V_WString
Size: 6
The_Hour (datetime)
1909-01-01 09:00:00
// This stips off the minutes/seconds
DateTimeFormat([Event_Date],'%Y-%m-%d %H:00:00')
Data type: DateTime
Size: 19
The Hour (str)
09
// Returns just the hour
// H = 0 to 23 hours
// I = 0 to 12 hours
DateTimeFormat([Event_Date],'%I')
Data type: V_String
Size: 4
The_Hour_(str am/pm)
AM
// This returns the hour
// p = am/pm
// P = AM/PM
DateTimeFormat([Event_Date],'%p')
Data type: V_String
Size: 4
The_Minute (datetime)
1909-01-01 09:53:00
// This stips off the seconds
DateTimeFormat([Event_Date],'%Y-%m-%d %H:%M:00')
Data type: DateTime
Size: 19
The_Minute (str)
53
// This returns the Minutes
// M = two digit Minutes
// p = am/pm
// P = AM/PM
DateTimeFormat([Event_Date],'%M')
Data type: V_String
Size: 4
HMS (time)
9:53:39 AM
// This returns the hours minutes seconds
// X = hours mins seconds am/pm
DateTimeFormat([Event_Date],'%X')
Data type: V_String
Size: 12



Converting Unix Timestamps (check your work at https://www.unixtimestamp.com/index.php)



Topic 4: Regex and Date Operations (Multiple weekly examples)

From Week 4 of the Weekly challenges

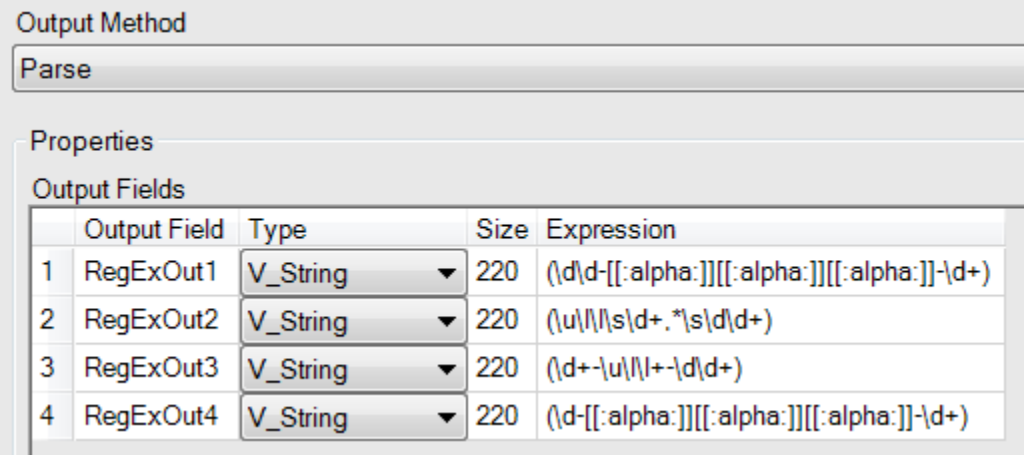
.*(\d\d-[[:alpha:]][[:alpha:]][[:alpha:]]-\d+).* | .*(\u\l\l\s\d+,*\s\d\d+).* | .*(\d+-\u\l\l+-\d\d+).* | .*(\d-[[:alpha:]][[:alpha:]][[:alpha:]]-\d+).*

There are four regex searches here -> and the example data that matches the search:

1. .*(\d\d-[[:alpha:]][[:alpha:]][[:alpha:]]-\d+).* -> 16-APR-2005
2. .*(\u\l\l\s\d+,*\s\d\d+).* -> Nov 16, 1900
3. .*(\d+-\u\l\l+-\d\d+).* -> 9-July-2001
4. .*(\d-[[:alpha:]][[:alpha:]][[:alpha:]]-\d+).* -> 4-SEP-00

Notice that the pipe (|) is used to delimit the searches and that ".*" is used at the beginning and the end of the searches to be able to find the 4 search patterns anywhere in the search area.

Alteryx creates 4 output fields sized at 220 to handle the content of the four searches, when the Parse method is used.



Example matches of these are:

Results - RegEx (54) - Output

5 of 5 Fields | Cell Viewer | 17 records displayed

Record #	Field 1	RegExOut1	RegExOut2	RegExOut3	RegExOut4
1	He who sleeps on the floor will not fall...	16-APR-2005			
2	After all is said and done, more is said t...	09-JAN-1856			
3	I want to see you shoot the way you sh...		Nov 16, 1900		
4	get someone else to do it.15-APR-1944...	15-APR-1944			
5	Why do they call it rush hour when not...	27-JUN-70			
6	I'm taking the Ryanair approach to it: s...	23-MAY-2011			
7	I Xeroxed a mirror. Now I have an extr...	30-JUN-06			
8	Freidrich Engels01-AUG-08This record i...	01-AUG-08			
9	'He's so old his social security number i...		Jan 5 2000		
10	"I was the best man at the wedding.So...			9-July-2001	
11	"When my wife was asked, "Do you tak...	21-May-07			
12	"These are the continuing voyagesTo b...	16-SEP-69			
13	"The best cure for insomnia is to get a l...				4-SEP-00

After some additional work using a formula tool,

Formula (56) - Configuration

Output Column | Data Preview

RegExOut1 | 16-APR-2005

```
if isempty([RegExOut1]) then (trim(substring([RegExOut2],4,2))+"-"+ Left(uppercase([RegExOut2]),3)+"-
"+right([RegExOut2],4)) else [RegExOut1] endif
```

Data type: V_String Size: 220

RegExOut1 | 16-APR-2005

```
if [RegExOut1]=="--" then [RegExOut3] else [RegExOut1] endif
```

Data type: V_String Size: 220

RegExOut1 | 16-APR-2005

```
if isempty([RegExOut1]) then [RegExOut4] else [RegExOut1] endif
```

Data type: V_String Size: 220

Results - Formula (56) - Output

5 of 5 Fields | Cell Viewer | 17 records displayed

Record #	Field 1	RegExOut1	RegExOut2	RegExOut3	RegExOut4
1	He who sleeps on the floor will not fall...	16-APR-2005			
2	After all is said and done, more is said t...	09-JAN-1856			
3	I want to see you shoot the way you sh...	16-NOV-1900	Nov 16, 1900		
4	get someone else to do it.15-APR-1944...	15-APR-1944			
5	Why do they call it rush hour when not...	27-JUN-70			
6	I'm taking the Ryanair approach to it: s...	23-MAY-2011			
7	I Xeroxed a mirror. Now I have an extr...	30-JUN-06			
8	Freidrich Engels01-AUG-08This record i...	01-AUG-08			
9	'He's so old his social security number i...	5-JAN-2000	Jan 5 2000		
10	"I was the best man at the wedding.So...	9-July-2001		9-July-2001	
11	"When my wife was asked, "Do you tak...	21-May-07			
12	"These are the continuing voyagesTo b...	16-SEP-69			
13	"The best cure for insomnia is to get a l...	4-SEP-00			4-SEP-00
14	I don't even butter my bread; I consider...	08-may-2003			
15	It matters not whether you win or lose;...	21-MAR-2005			
16	Smoking is one of the leading causes o...	24-OCT-1989			
17	I tried to think but nothing happened!"...	11-AUG-1935			

And a text to columns parse:

&Y

Text To Columns (57) - Configuration

Field to Split: RegExOut1

Delimiters: -

Split to Columns

# of Columns: 3

Extra Columns: Leave Extra in Last Field

Output Root Name: Date

Split to Rows

Advanced Options

- Ignore Delimiters in Quotes
- Ignore Delimiters in Single Quotes
- Ignore Delimiters in Parenthesis
- Ignore Delimiters in Brackets
- Skip Empty Fields

Results - Text To Columns (57) - Output

8 of 8 Fields | Cell Viewer | 17 records displayed

Record #	Field 1	ReqExOut1	ReqExOut2	ReqExOut3	ReqExOut4	Date1	Date2	Date3
1	He who sleeps on the floor will not fall...	16-APR-2005				16	APR	2005
2	After all is said and done, more is said t...	09-JAN-1856				09	JAN	1856
3	I want to see you shoot the way you sh...	16-NOV-1900	Nov 16, 1900			16	NOV	1900
4	get someone else to do it.15-APR-1944...	15-APR-1944				15	APR	1944
5	Why do they call it rush hour when not...	27-JUN-70				27	JUN	70
6	I'm taking the Ryanair approach to it: s...	23-MAY-2011				23	MAY	2011
7	I Xeroxed a mirror. Now I have an extr...	30-JUN-06				30	JUN	06
8	Freidrich Engels01-AUG-08This record i...	01-AUG-08				01	AUG	08
9	'He's so old his social security number i...	5-JAN-2000	Jan 5 2000			5	JAN	2000
10	"I was the best man at the wedding.So...	9-July-2001		9-July-2001		9	July	2001
11	"When my wife was asked, "Do you tak...	21-May-07				21	May	07
12	"These are the continuing voyagesTo b...	16-SEP-69				16	SEP	69
13	"The best cure for insomnia is to get a l...	4-SEP-00			4-SEP-00	4	SEP	00
14	I don't even butter my bread; I consider...	08-may-2003				08	may	2003
15	It matters not whether you win or lose;...	21-MAR-2005				21	MAR	2005
16	Smoking is one of the leading causes o...	24-OCT-1989				24	OCT	1989
17	I tried to think but nothing happened!"...	11-AUG-1935				11	AUG	1935

the final dates are assembled using the DateTimeParse function:

DateTime_Out 2005-04-16

```
DateTimeParse(([Date1]+"-"+[Date2]+"-"+[Date3]),"%d-%b-%Y")
```

Results - Formula (58) - Output

9 of 9 Fields | Cell Viewer | 17 records displayed | Data | Metadata

Record #	Field 1	ReqExOut1	ReqExOut2	ReqExOut3	ReqExOut4	Date1	Date2	Date3	DateTime Out
1	He who sleeps on the floor will not fall...	16-APR-2005				16	Apr	2005	2005-04-16
2	After all is said and done, more is said t...	09-JAN-1856				09	Jan	1856	1856-01-09
3	I want to see you shoot the way you sh...	16-NOV-1900	Nov 16, 1900			16	Nov	1900	1900-11-16
4	get someone else to do it.15-APR-1944...	15-APR-1944				15	Apr	1944	1944-04-15
5	Why do they call it rush hour when not...	27-JUN-70				27	Jun	1970	1970-06-27
6	I'm taking the Ryanair approach to it: s...	23-MAY-2011				23	May	2011	2011-05-23
7	I Xeroxed a mirror. Now I have an extr...	30-JUN-06				30	Jun	2006	2006-06-30
8	Freidrich Engels01-AUG-08This record i...	01-AUG-08				01	Aug	2008	2008-08-01
9	'He's so old his social security number i...	5-JAN-2000	Jan 5 2000			05	Jan	2000	2000-01-05
10	"I was the best man at the wedding.So...	9-July-2001		9-July-2001		09	Jul	2001	2001-07-09
11	"When my wife was asked, "Do you tak...	21-May-07				21	May	2007	2007-05-21
12	"These are the continuing voyagesTo b...	16-SEP-69				16	Sep	1969	1969-09-16
13	"The best cure for insomnia is to get a l...	4-SEP-00			4-SEP-00	04	Sep	2000	2000-09-04

For Reference, here are the specifiers used for dates/time for Alteryx:

Specifiers always begin with a percent sign (%), followed by a case-sensitive letter. The data must include at least a two digit year.

Specifier	Output from DateTimeFormat	Supported Input with DateTimeParse
%a	Abbreviated weekday name ("Mon")	Any valid abbreviation of a day of the week ("mon", "Tues.", "Thur"), giving an error only if the text given is not a day of the week. Note that Alteryx does not check that the specified day name is valid for a particular date.
%A	Full weekday name ("Monday")	Day name or any valid abbreviation of a day of the week ("mon", "Tues.", "Thur"), giving an error only if the text given is not a day of the week. Note that Alteryx does not check that the specified day name is valid for a particular date.
%b	Abbreviated month name ("Sep")	Any valid abbreviation of a month name ("Sep", "SEPT."), giving an error only if the text given is not a name of a month.
%B	Full month name ("September")	Month name or any valid abbreviation of a month name ("Sep", "SEPT."), giving an error only if the text given is not a name of a month.
%c	The date and time for the computer's locale	Not supported
%C	The century number ("20")	Not supported
%d	Day of the month ("01")	One or two digits, ignoring spaces ("1" or "01")
%D	Equivalent to %m/%d/%y	Not supported
%e	Day of the month, leading 0 replaced by a space (" 1")	One or two digits, ignoring spaces ("1" or "01")
%h	Same as %b ("Sep")	Any valid abbreviation of a month name ("Sep", "SEPT."), giving an error only if the text given is not a name of a month.
%H	Hour in 24 hour clock, 00 to 23	Up to two digits for hour, 0 to 23. Not compatible with **%p** or **%P**.
%I (capital "eye")	Hour in 12 hour clock, 01 to 12	Up to two digits for hour, 1 to 12. Must follow with **%p** or **%P**.
%j	The day of the year, from 001 to 365 (or 366 in leap years)	3-digit day of the year, from 001 to 365 (or 366 in leap years)
%k	24 hours, leading zero is space, " 0" to "23"	Up to two digits for hour
%l (lowercase "ell")	12 hours, leading zero is space, " 1" to "12"	Not supported
%M	Minutes, 00 to 59	Up to two digits for minutes
%m	Month number, 01 to 12	One or two digit month number, 1 or 01 to 12
%p	"AM" or "PM"	Case blind ("aM" or "Pm"). Must follow **%I** (capital "eye", hour in 12-hour format)
%P	"am" or "pm"	Case blind ("aM" or "Pm"). Must follow **%I** (capital "eye", hour in 12-hour format)
%S	Seconds, 00 to 59	Up to two digits for seconds
%T	Time in twenty-four hour notation. Equivalent to %H:%M:%S	Not supported
%u	Day of week as a decimal, 1 to 7, with Monday as 1	Not supported
%U	This returns the week number, as 00 – 53, with the beginning of weeks as Sunday.	Not supported
%w	Day of week as a number, 0 to 6, with Sunday as 0	Not supported
%W	This returns the week number, as 00 – 53, with the beginning of weeks as Monday.	Not supported
%x	The date for the computer's locale	Not supported
%X	The 12-hour clock time, including AM or PM ("11:51:02 AM")	Hours:Minutes:Seconds [AM / PM]
%y	Last two digits of the year ("16")	Up to four digits are read, stopping at a separator or the end of the string, and mapped to a range of the current year minus 66 to current year plus 33. (For example, in 2016, that's 1950 to 2049.) › Limitation with six-digit dates
%Y	All four digits of the year ("2016")	Two or four digits are read. Two digits are mapped to a range of the current year minus 66 to current year plus 33. (For example, in 2016, that's 1950 to 2049.)
%z	Offset from UTC time ("-600")	Not supported
%Z	Full timezone name ("Mountain Daylight Time")	Not supported

The separators:

Separators

Separators are inserted between date/time specifiers to form a format string.

Separator	Output from DateTimeFormat	Supported Input with DateTimeParse*
/	/	/ or -
-	-	/ or -
space	A space	Any sequence of white space characters
%n	A newline	Not supported
%t	A tab	Not supported
other	Other characters, such as comma, period, and colon	Other characters, such as comma, period, and colon

* DateTimeParse accepts forward slashes (/) and hyphens (-) interchangeably. However, commas, colons, and all other separators must match the incoming data exactly.

And the Date/Time Examples:

Format string examples

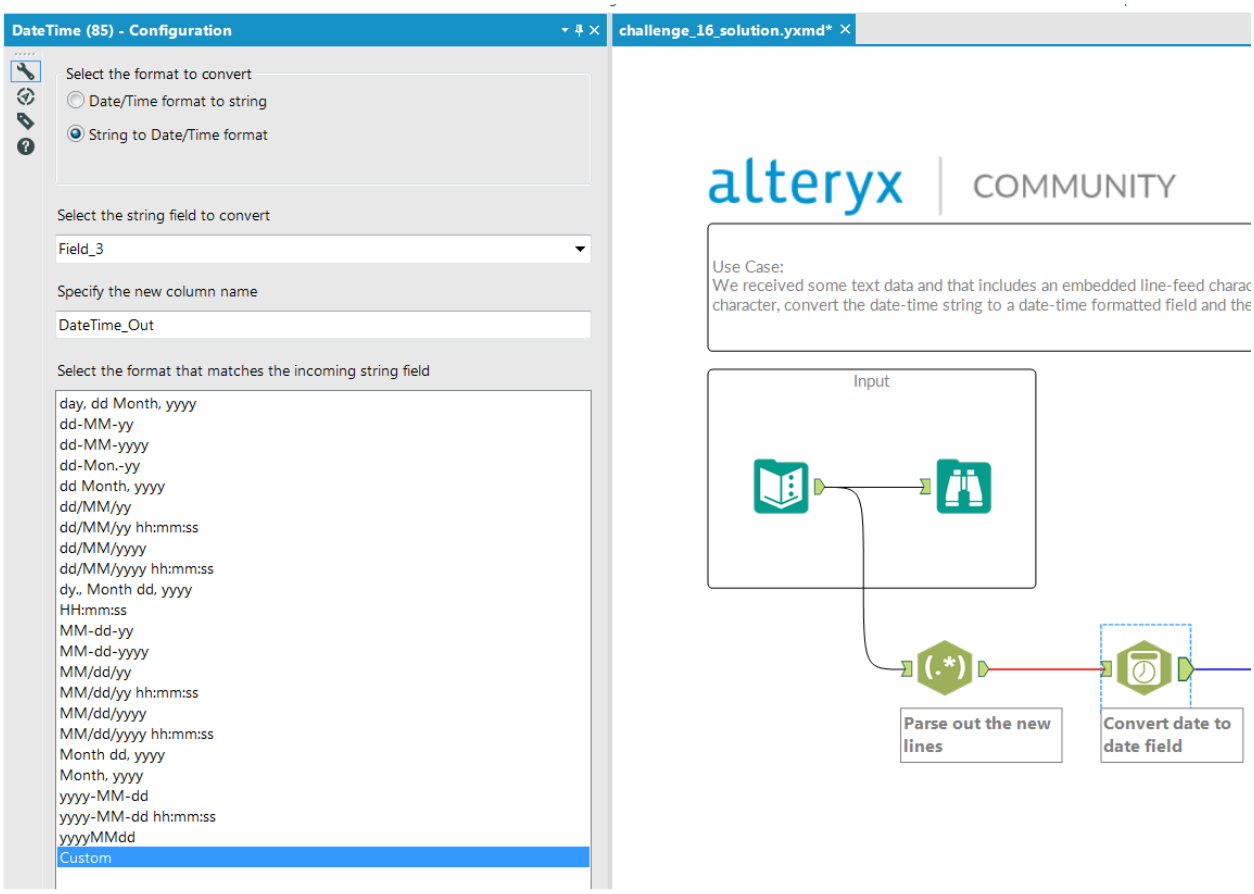
Format String	Result
`%d-%b-%y`	01-Aug-16
`%A, %d %B, %Y`	Monday, 01 August, 2016
`%d-%m-%y`	01-08-16
`%d-%m-%Y`	01-08-2016
`%d %B, %Y`	01 August, 2016
`%d/%m/%y`	01/08/16
`%d/%m/%Y`	01/08/2016
`%a, %B %d, %Y`	Mon, August 01, 2016
`%A, %B%e, %Y`	Monday, August 1, 2016
`%m-%d-%y`	08-01-16
`%m-%d-%Y`	08-01-2016
`%m/%d/%y`	08/01/16
`%m/%d/%Y`	08/01/2016
`%b %d`	Aug 01
`%B %d, %Y`	August 01, 2016
`%B, %Y`	August, 2016
`%Y-%m-%d`	2016-08-01
`%Y%m%d`	20160801

To find a match for anything:

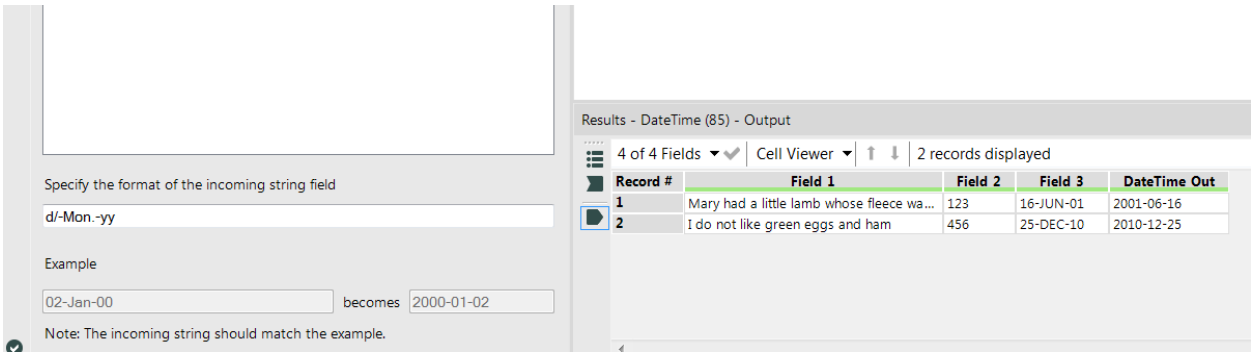
(.?)*

Datetime Tool Example 1: Custom format date string

From Week 16, a custom formatted string (16-JUN-01) is converted to a date (2001-06-16) using the datetime tool.

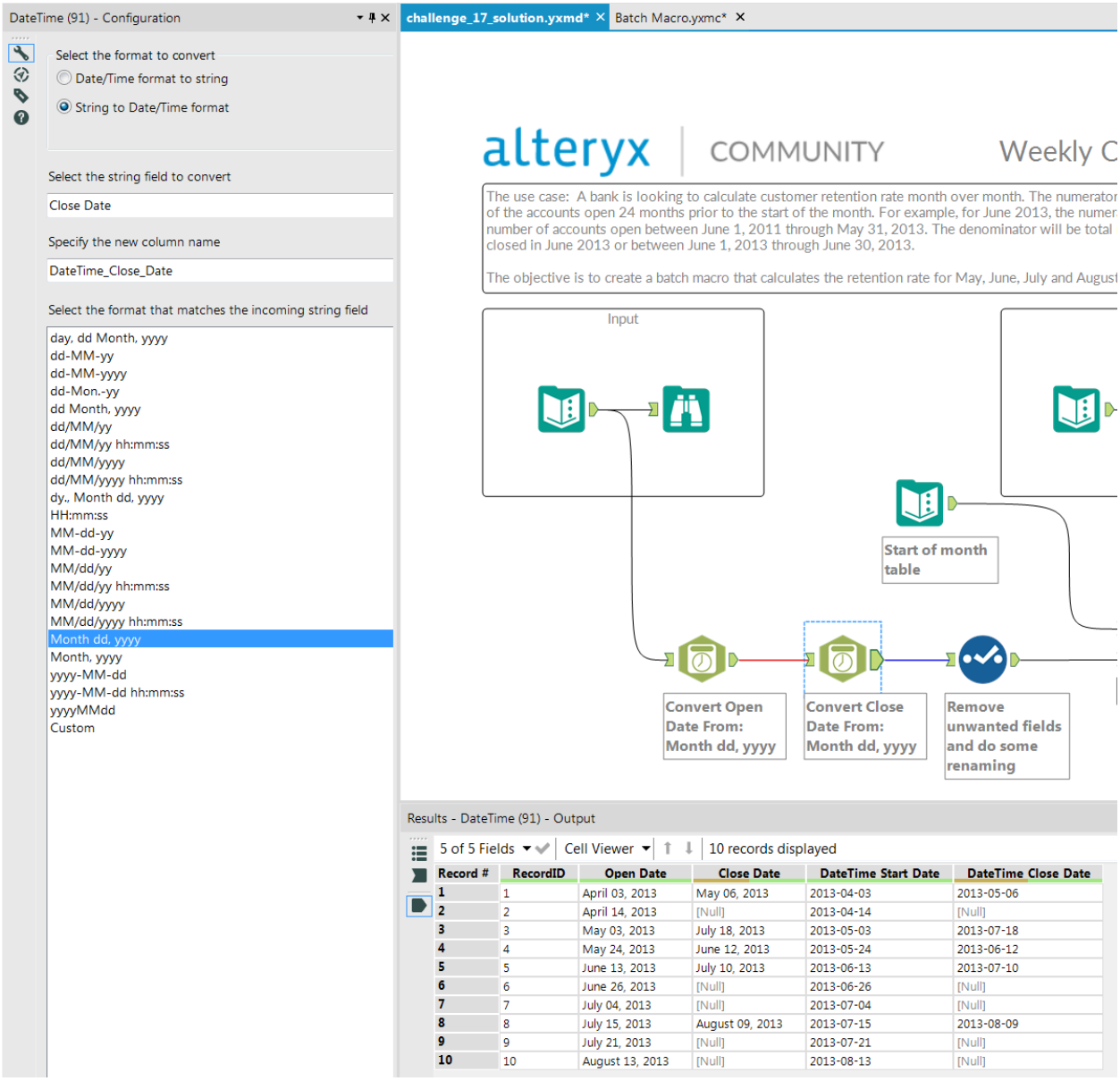


The custom setting is shown below as d/-Mon.-yy .When this is used, Field 3 becomes a DateTime Out.



Datetime Tool Example 2: Standard format date string

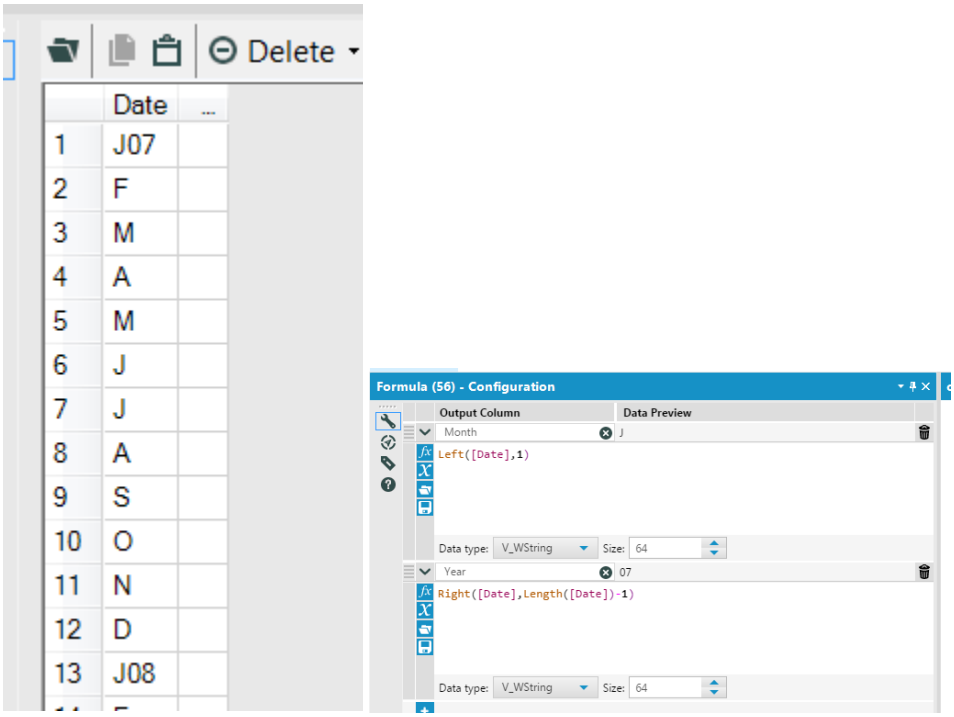
From Week 17, a standard formatted string (April 03, 2013) is converted to a date (2013-04-03) using the datetime tool.



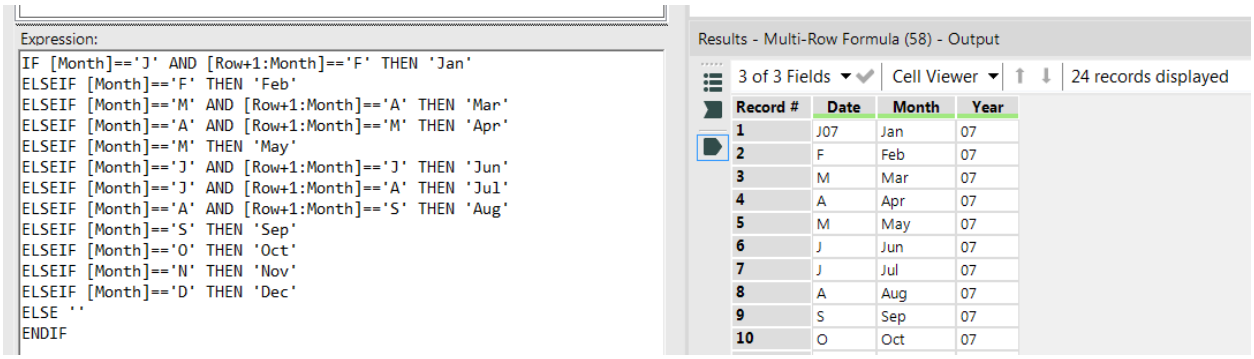
Record #	RecordID	Open Date	Close Date	DateTime Start Date	DateTime Close Date
1	1	April 03, 2013	May 06, 2013	2013-04-03	2013-05-06
2	2	April 14, 2013	[Null]	2013-04-14	[Null]
3	3	May 03, 2013	July 18, 2013	2013-05-03	2013-07-18
4	4	May 24, 2013	June 12, 2013	2013-05-24	2013-06-12
5	5	June 13, 2013	July 10, 2013	2013-06-13	2013-07-10
6	6	June 26, 2013	[Null]	2013-06-26	[Null]
7	7	July 04, 2013	[Null]	2013-07-04	[Null]
8	8	July 15, 2013	August 09, 2013	2013-07-15	2013-08-09
9	9	July 21, 2013	[Null]	2013-07-21	[Null]
10	10	August 13, 2013	[Null]	2013-08-13	[Null]

Week 21 – More Custom Date Work

In this example, very sketchy date details are provided and complete month/years are created from the information. Here is the initial sketchy data followed by the parsing of month and year.



Here is the final date output, showing the clever logic used to rename the months:



Topic 5: Multifield searching and matching (Week 5)

The append tool is used to create combinations of an input value and records in a database such that the input field can be found in any of the columns of the database. The append operation creates the combinations needed for this to be possible, and a simple if block does the comparisons.

Results - Append Fields (67) - Output

5 of 5 Fields | Cell Viewer | 19 records displayed

Record #	Position Number	Level0	Level1	Level2	Level3
1	3333	123456	[Null]	[Null]	[Null]
2	3333	123456	111111	[Null]	[Null]
3	3333	123456	111111	22222	[Null]
4	3333	123456	111111	22222	33333
5	3333	123456	111111	23333	[Null]
6	3333	123456	111111	23333	34444
7	3333	123456	111111	23333	35555
8	3333	123456	12222	[Null]	[Null]
9	3333	123456	12222	234444	[Null]
10	3333	123456	12222	234444	366666
11	3333	123456	12222	33333	36677
12	3333	123456	12222	234444	37777
13	3333	123456	12222	[Null]	[Null]
14	3333	123456	12222	235555	[Null]
15	3333	123456	12222	235555	388888
16	3333	123456	12222	235555	399999
17	3333	123456	12222	235555	399888
18	3333	123456	12222	235555	3998877
19	3333	123456	33333	235555	388888

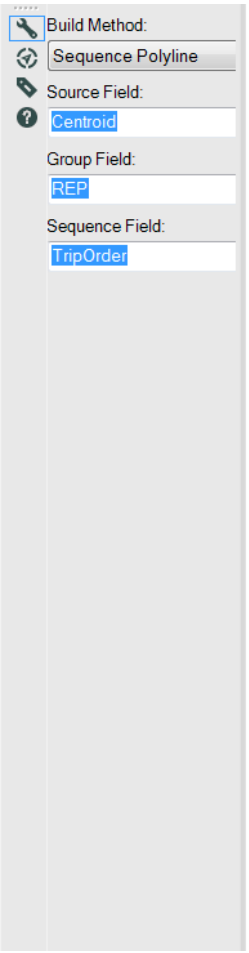
The user input of 3333 is appended to the database records. The following logic identifies the records where 3333 is found.

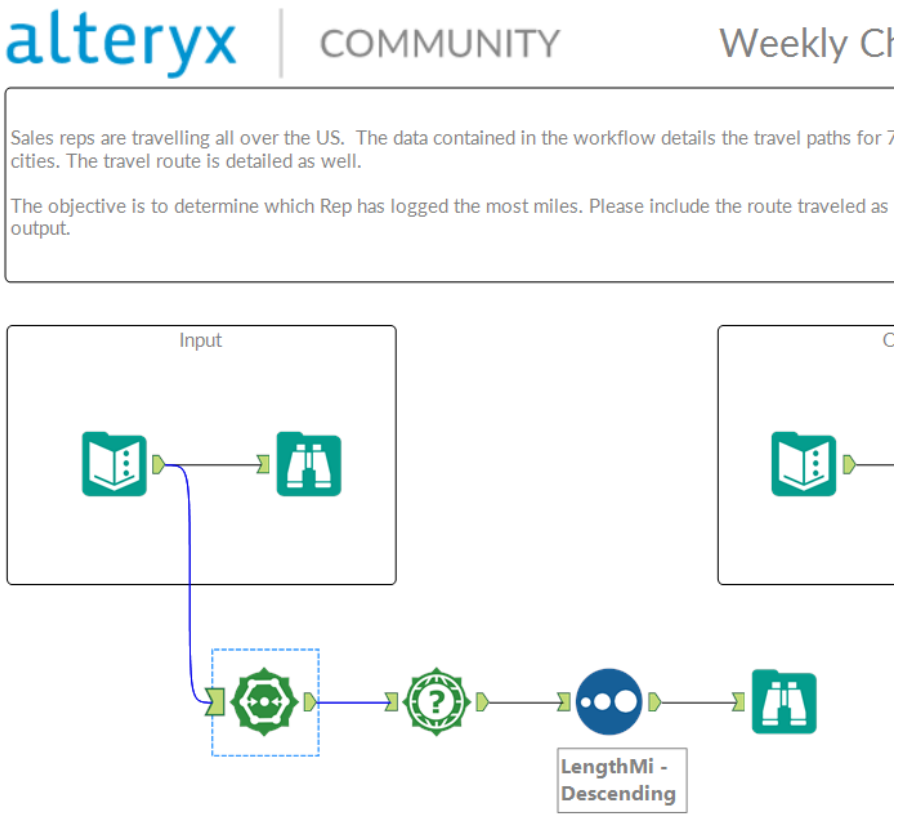
Expression:

```
[Position Number]==[Level0] or
[Position Number]==[Level1] or
[Position Number]==[Level2] or
[Position Number]==[Level3]
```

Topic 6: Length along a Polyline (Week 6)

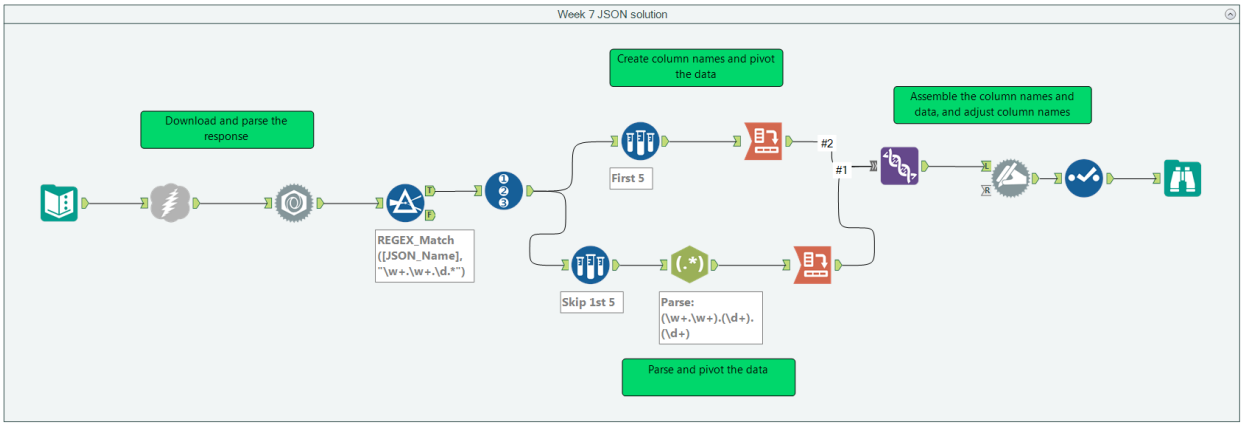
A sequence of airport trips are strung together to find out which sales rep as traveled the most miles. The airport lat/longs are given as centroids so all that is necessary is to produce polylines for each sales rep and use the spatial info tool to calculate the distance traveled by each sales rep.





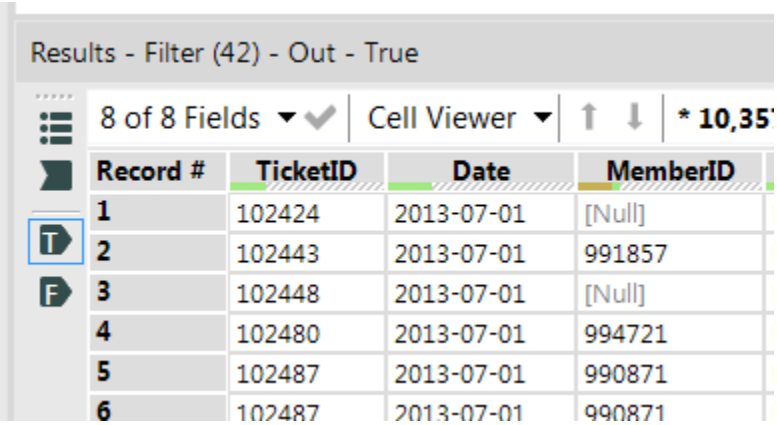
Topic 7: Parsing JSON Data (Week 7)

This is an excellent example in so many ways. The methods used to identify the JSON data elements are insightful and efficient. There are so many excellent maneuvers in this example that it is one of the best exercises to date. I have rarely used the sample tool, and it is used in two different ways here. I have never used the JSON tool, so it was good to learn. Finally, the use of regex and the dynamic rename tool were both good.



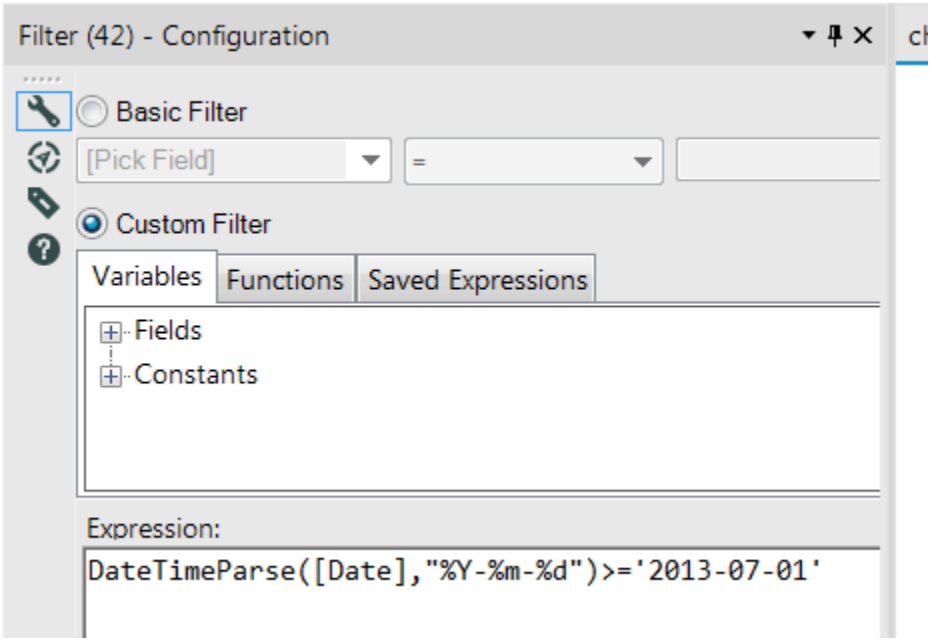
Topic 8: Filtering by date (week 8)

Given date data like:

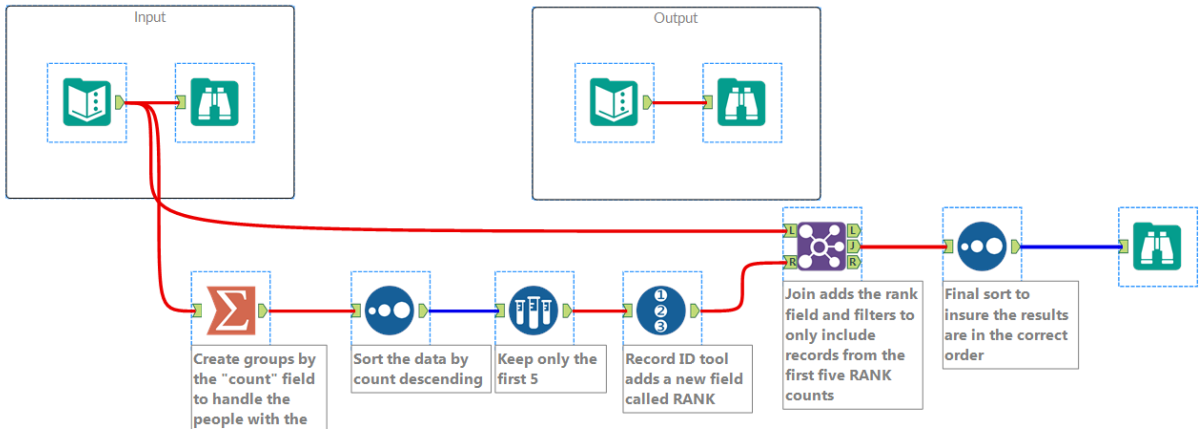


Record #	TicketID	Date	MemberID
1	102424	2013-07-01	[Null]
2	102443	2013-07-01	991857
3	102448	2013-07-01	[Null]
4	102480	2013-07-01	994721
5	102487	2013-07-01	990871
6	102487	2013-07-01	990871

Configure a filter to allow date-based filtering



Topic 9: Ranking items where there can be more than 1 at the same rank level, and performing a top N calculation (Week 9)



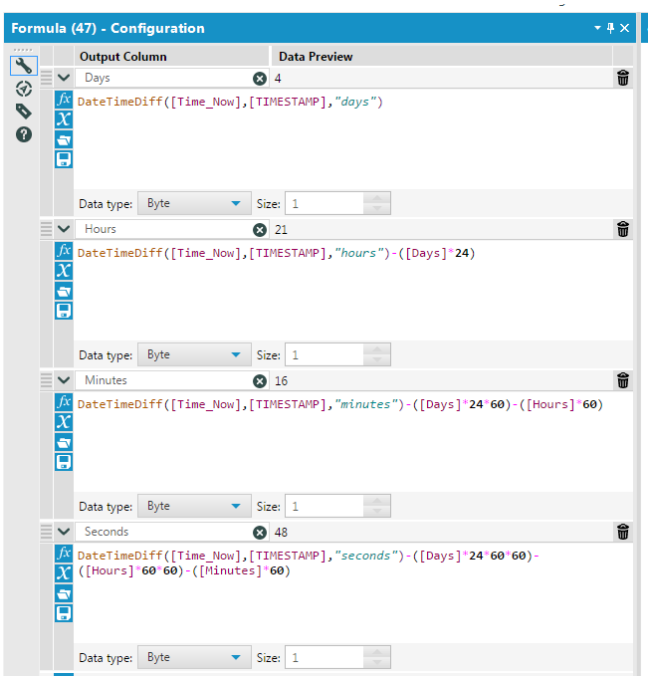
I like this example because of the use of the sample tool to identify the top N ranks, and also for the use of the clever technique used to assign the ranks (using a join).

Topic 10: Calculating Time (Days, hours, minutes, seconds)

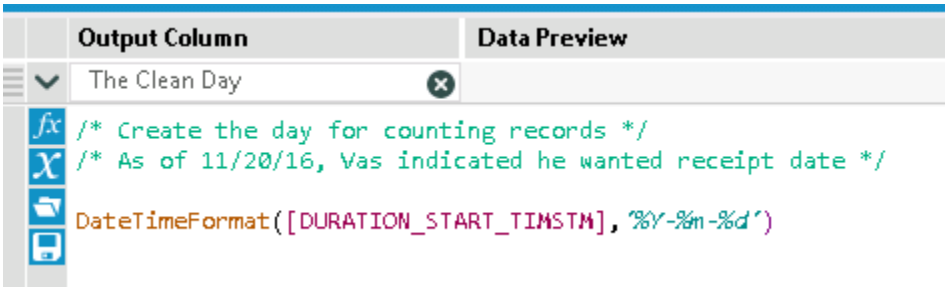
Has an error in the naming of the first formula. This says it is a time difference in minutes but is actually a difference in seconds. Otherwise, excellent instructional on how to calculate discrete time blocks.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Registrant	TIMESTAMP	Time_Now	TimeDiffMinutes	Days	Hours	Minutes	Seconds		Days Hours Minutes Seconc		Minutes Calculated	Seconds Calculated
2	HPNZGSDN	7/9/2014 11:07	7/14/2014 8:24	422208	4	21	16	48		4 21:16:48		7036	422208
3	F5NZRZ3Y	7/9/2014 8:40	7/14/2014 8:24	431068	4	23	44	28		4 23:44:28			
4	FHNBTNM	7/8/2014 12:26	7/14/2014 8:24	503859	5	19	57	39		5 19:57:39			
5	ZHN7W97	7/8/2014 13:26	7/14/2014 8:24	500277	5	18	57	57		5 18:57:57			
6	ZKNWRVB	7/8/2014 13:25	7/14/2014 8:24	500333	5	18	58	53		5 18:58:53			
7	HGNYD3V	7/7/2014 19:13	7/14/2014 8:24	565871	6	13	11	11		6 13:11:11			

For a more efficient solution, see the following formulas



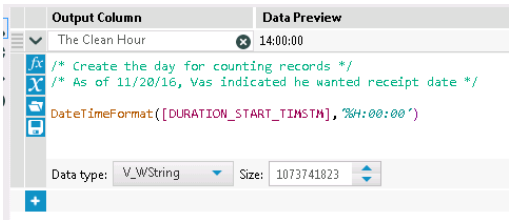
Creating a day bucket from a datetime field



/* Create the day for counting records */

DateTimeFormat([DURATION_START_TIMSTM],'%Y-%m-%d')

Creating an hour bucket



DateTimeFormat([DURATION_START_TIMSTM],'%H:00:00')

The Minute Bucket With the day:

DateTimeFormat([DURATION_START_TIMSTM],'%Y-%m-%d %H:%M:00')

The Minute Bucket:

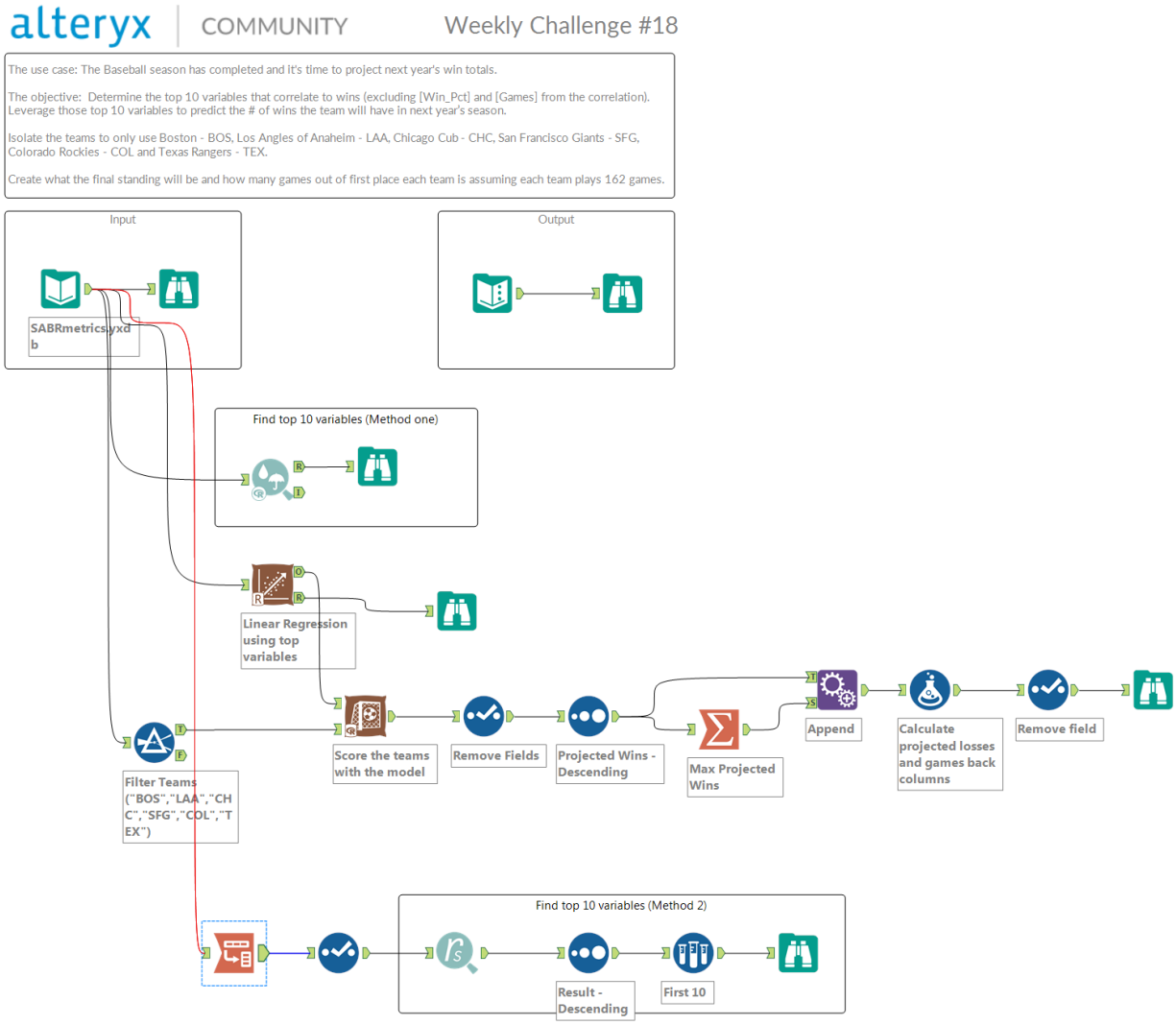
DateTimeFormat([CALL_START_DT],'%H:%M:00')

You can also create an hourly bucket for all days of your data like this:

DateTimeFormat([DURATION_START_TIMSTM],'%Y-%m-%d %H:00:00')

With this formulation, you will get 24 records per day times the number of days you have in the file.

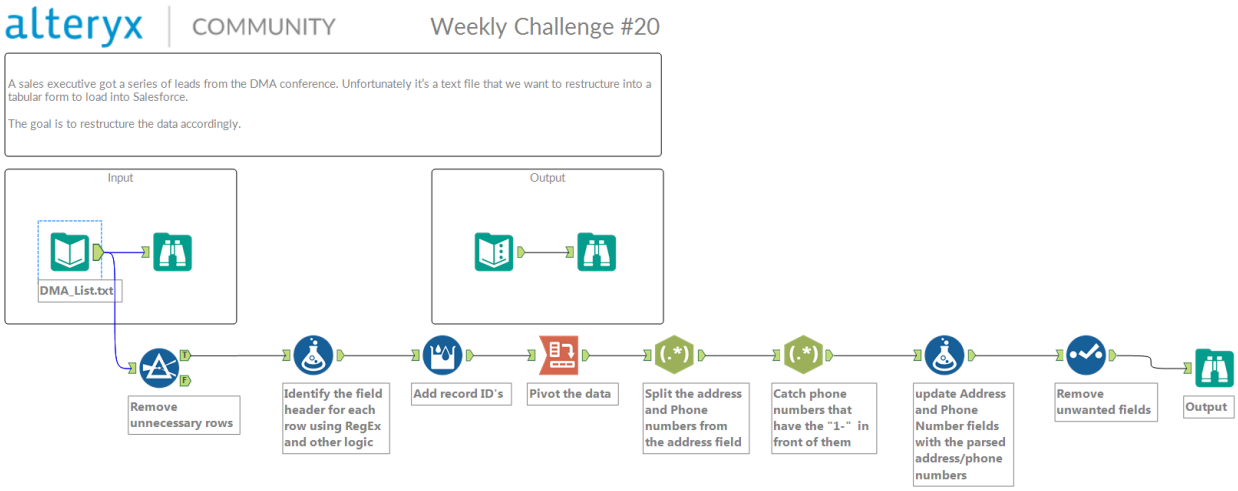
Topic 11: Linear Regression Modeling



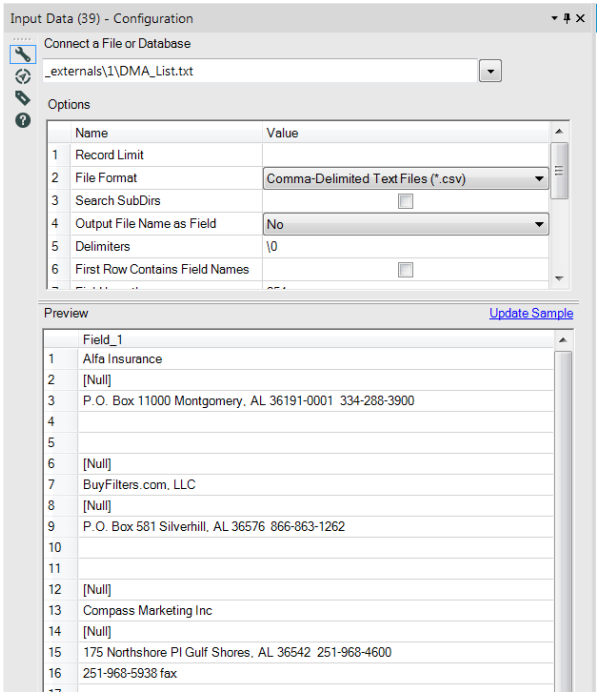
I like this example because it uses the Spearman Correlation tool to identify the top 10 statistics that are most strongly correlated to winning baseball games (lower part of workflow) than then these terms are used in a linear regression model to estimate how teams will do in the following season. I especially like the use of the scoring tool to determine the teams which are best positioned to win the following year. It would be an interesting study to take historical data, apply this approach and see how accurate the results were. I'd like to do the same thing for football.

Topic 12: Identifying Data Fields in Sloppy Data

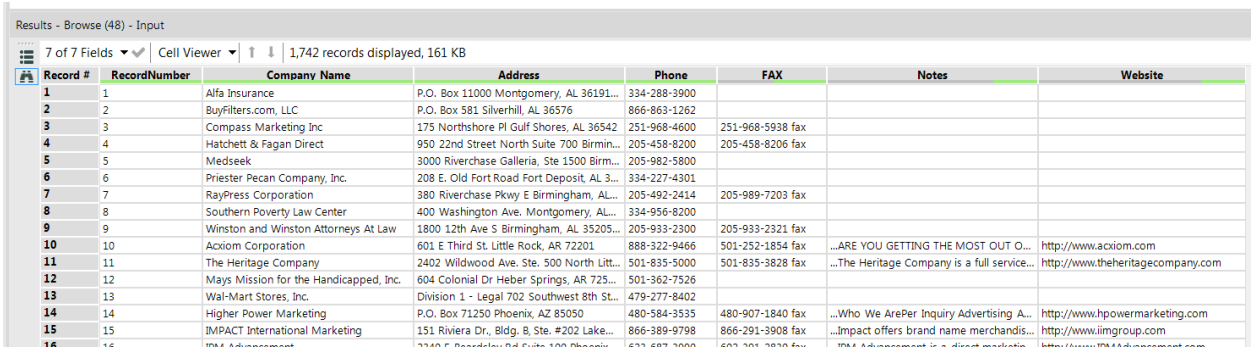
This is example 20 and I like it a lot because of how regex parsing is used to identify different data type elements like addresses, phone numbers, etc. The buckets are created to hold these fields and I think the approach is novel and robust. There are many real-work examples that could use this approach.



The incoming data looks like this:



Once the cleaning and parsing is complete, a nice output structure is achieved:



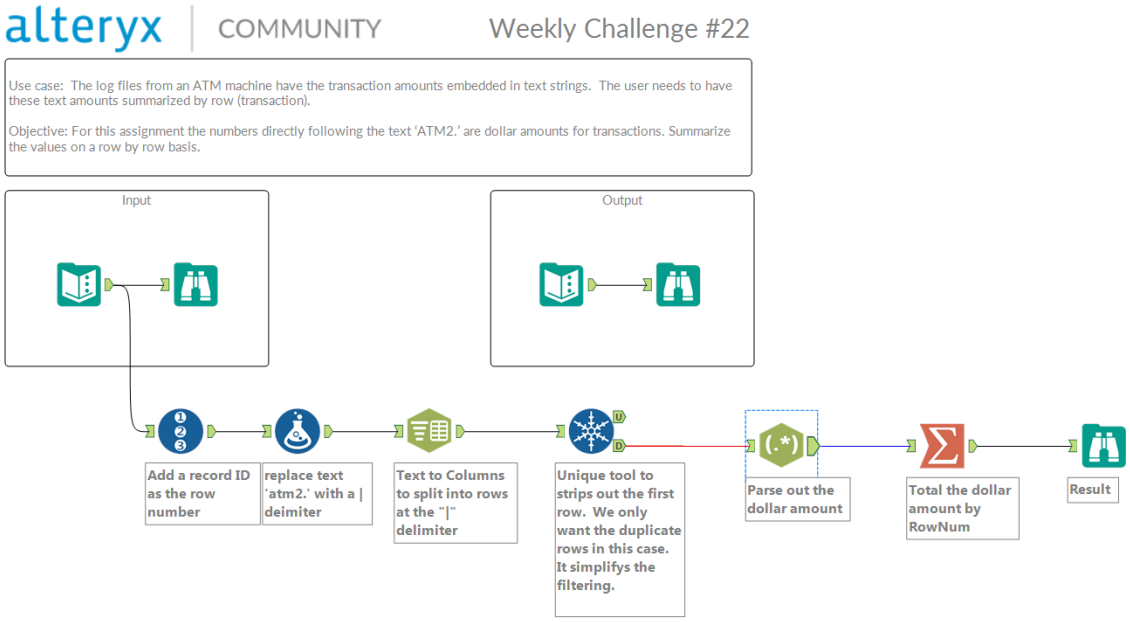
Results - Browse (48) - Input

7 of 7 Fields | Cell Viewer | 1,742 records displayed, 161 KB

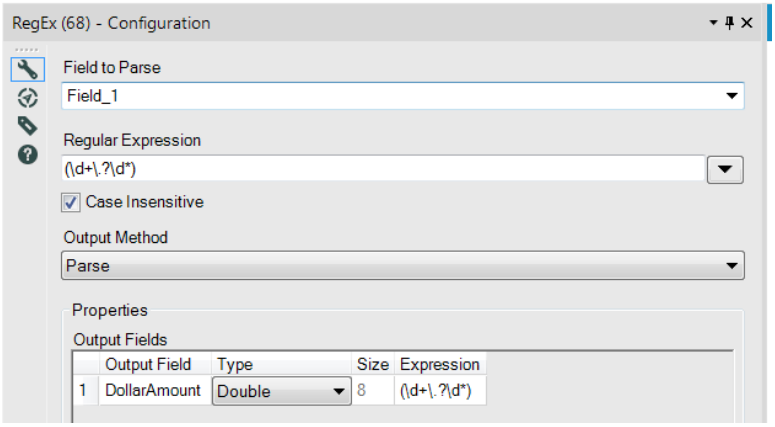
Record #	RecordNumber	Company Name	Address	Phone	FAX	Notes	Website
1	1	Alfa Insurance	P.O. Box 11000 Montgomery, AL 36191...	334-288-3900			
2	2	BuyFilters.com, LLC	P.O. Box 581 Silverhill, AL 36576	866-863-1262			
3	3	Compass Marketing Inc	175 Northshore Pl Gulf Shores, AL 36542	251-968-4600	251-968-5938 fax		
4	4	Hatchett & Fagan Direct	950 22nd Street North Suite 700 Birmin...	205-458-8200	205-458-8206 fax		
5	5	Medseek	3000 Riverchase Galleria, Ste 1500 Birm...	205-982-5800			
6	6	Priester Pecan Company, Inc.	208 E. Old Fort Road Fort Deposit, AL 3...	334-227-4301			
7	7	RayPress Corporation	380 Riverchase Pkwy E Birmingham, AL...	205-492-2414	205-989-7203 fax		
8	8	Southern Poverty Law Center	400 Washington Ave. Montgomery, AL...	334-956-8200			
9	9	Winston and Winston Attorneys At Law	1800 12th Ave S Birmingham, AL 35205...	205-933-2300	205-933-2321 fax		
10	10	Acxiom Corporation	601 E Third St. Little Rock, AR 72201	888-322-9466	501-252-1854 fax	...ARE YOU GETTING THE MOST OUT O...	http://www.acxiom.com
11	11	The Heritage Company	2402 Wildwood Ave. Ste. 500 North Litt...	501-835-5000	501-835-3828 fax	...The Heritage Company is a full service...	http://www.theheritagecompany.com
12	12	Mays Mission for the Handicapped, Inc.	604 Colonial Dr Heber Springs, AR 725...	501-362-7526			
13	13	Wal-Mart Stores, Inc.	Division 1 - Legal 702 Southwest 8th St...	479-277-8402			
14	14	Higher Power Marketing	P.O. Box 71250 Phoenix, AZ 85050	480-584-3535	480-907-1840 fax	...Who We ArePer Inquiry Advertising A...	http://www.hpowermarketing.com
15	15	IMPACT International Marketing	151 Riviera Dr., Bldg. B, Ste. #202 Lake...	866-389-9798	866-291-3908 fax	...Impact offers brand name merchandis...	http://www.iimgroup.com
16	16	IPM Advancement	2240 E Beardsley Rd Suite 100 Phoenix	623-687-2000	602-391-2820 fax	IPM Advancement is a direct marketin...	http://www.IPMAdvancement.com

Here are the details of how the data fields are identified: (Awesome regex examples)

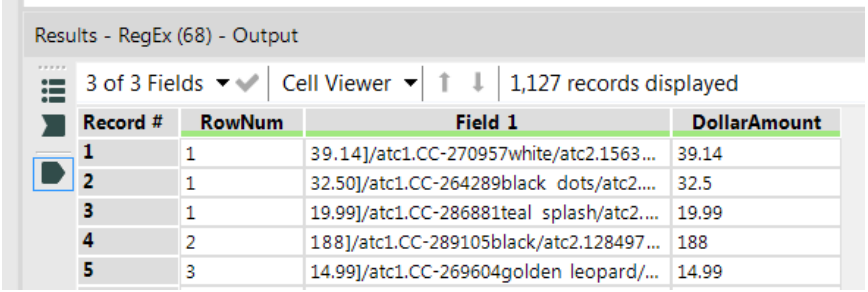
Continuing with the theme of sloppy data, Week 22 has ATM data in a really ugly format and the dollar transactions need to be extracted. This is another nice regex example. Here is the workflow:



Here is **the regex for extracting the dollar values of the transactions**:

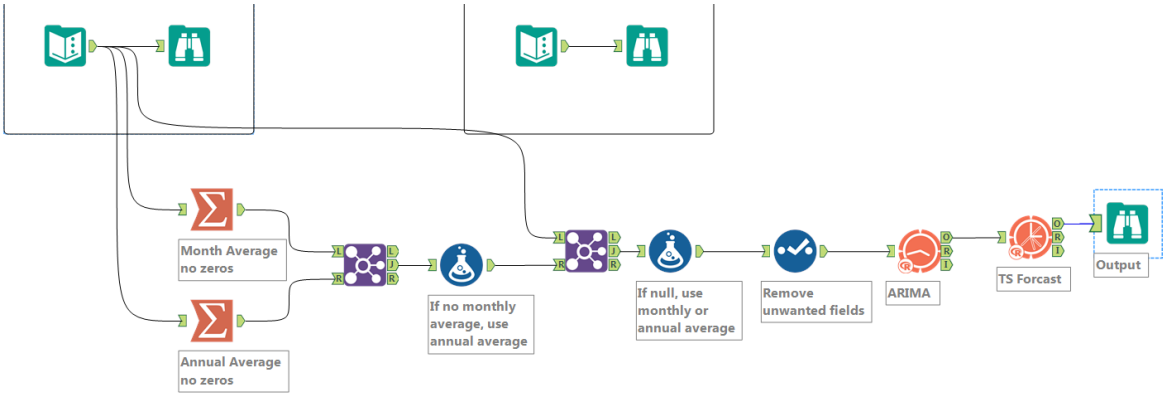


Here is the result:



Record #	RowNum	Field 1	DollarAmount
1	1	39.14]/atc1.CC-270957white/atc2.1563...	39.14
2	1	32.50]/atc1.CC-264289black dots/atc2....	32.5
3	1	19.99]/atc1.CC-286881teal splash/atc2....	19.99
4	2	188]/atc1.CC-289105black/atc2.128497...	188
5	3	14.99]/atc1.CC-269604golden leopard/...	14.99

Topic 13: Time Series Forecasting Using An autoregressive integrated moving average (ARIMA) model



Results - Browse (104) - Input

7 of 7 Fields | Cell Viewer | 6 records displayed, 1853 bytes

Record #	Period	Sub Period	forecast	forecast high 95	forecast high 80	forecast low 80	forecast low 95
1	3	1	334.738227	467.824761	421.758834	247.717621	201.651694
2	3	2	293.126438	459.723995	402.058725	184.19415	126.528881
3	3	3	261.793006	444.683601	381.378738	142.207273	78.90241
4	3	4	238.199116	429.713072	363.423361	112.97487	46.685159
5	3	5	220.433055	416.668187	348.744311	92.1218	24.197924
6	3	6	207.055317	405.917568	337.084354	77.02628	8.193065

I really like this example for a few different reasons. Using Alteryx to make predictions is a very practical usage of the software. I especially like the forecasting at 95% and 80% high and low.

Miscellaneous Notes

About the Text to Columns tool

The text to columns tool takes the text in one column and splits the string value into separate, multiple columns (or rows), based on a single or multiple delimiter (s).

▶ Watch the video

Configuration Properties

1. Specify the field to split. Choose this field from the drop down list.
2. Specify the delimiter (s) to use to split the data on. You can type the character in this space (i.e. **&**, :, ',etc.) or use the following:

   \t = Tab

   \n = New line

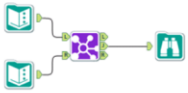
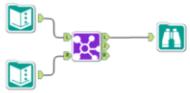
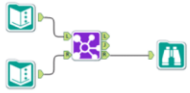
   \s = Space

   *Note: If you wanted to separate a field using a space OR tab, you would put \s\t in the delimiter. This is especially handy with the **Skip Empty Fields** in the Advanced Options.*
3. Choose your method for splitting:

   ▶ **Split to Columns:** will split a single column of data at each instance of the specified delimiter into multiple columns.
   ▶ **Split to Rows:**will split a single column of data at each instance of the specified delimiter into multiple rows.

Doing different types of Joins

See the table below for using the Join tool to execute different types of joins.

Inner Join:contains records that joined from the L input to those records in the R input.		
Left Unjoin: contains records from the L input that did NOT join to records from the R input.		
Right Unjoin: contains records from the R input that did NOT join to records from the L input.		
Left Outer Join: all records from the L input including the records that joined with the R input.		
Right Outer Join: all records from the R input including the records that joined with the L input.		
Full Outer Join: all of the records from both L and R inputs.		

The J output of the Join tool contains the result of an Inner Join.
The L output of the Join tool contains the result of a Left Unjoin.
The R output of the Join tool contains the result of a Right Unjoin.
To do a Left Outer Join, connect the J and L outputs of the Join tool to the Union tool. Connect the J output first to establish the combined table schema.
To do a Right Outer Join, connect the J and R outputs of the Join tool to the Union tool. Connect the J output first to establish the combined table schema.
To do a Full Outer Join, connect the J, L, and R outputs of the Join tool to the Union tool. Connect the J output first to establish the combined table schema.

Data Field Types

TYPE	DESCRIPTION	
Bool	Boolean: The type of an expression with two possible values: True or False	0=False; -1=True Note: any value other than 0 would indicate the value is True.
Byte	Number: A byte field is a positive whole number than falls within the range, 0 thru 255	0, 1, 2, 3....253, 254, 255
Int16	Number: 2Byte: Twice Exponential to the Byte, or 2^{16}	–32,768 to 32,767
Int32	Number: 4Byte: Four Times Exponential to the Byte, or 2^{32}	–2,147,483,648 to 2,147,483,647
Int64	Number: 8Byte: Eight Times Exponential to the Byte, or -2^{63} to $+2^{63}$	–9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
Fixed Decimal	Number: The specification of width of field and then to decimal threshold. The first number is the total width of number, the second number is to the decimal level of precision. The decimal point is included in the character width.	"7.2" => 1234.56 "8.2" => -1234.56
Float	Number: A single-precision floating point number is a 32-bit approximation of a real number.	+/- 3.4E +/- 38 (7 digits) where 38 is the exponent and 7 digits references seven digits of accuracy.
Double	Number: A double-precision floating point number is a 64-bit approximation of a real number.	+/- 1.7E +/- 308 (15 digits) where 308 is the exponent and 15 digits references fifteen digits of accuracy.
String	Character: Fixed Length String. The length must be at least as large as the largest character value contained in the field. Limited to 8192 characters.	Any string whose length does not vary much from value to value.
V_String	Character: Variable Length. Length of field will adjust to accommodate the entire string within the field.	If the string greater than 16 characters and varies in length from value to value
WString	Character: Wide String will accept unicode characters. Limited to 8192 characters.	Any string whose length does not vary much from value to value. Æ.ç.ß..Ð.Ñ... Any string that contains unicode characters
V_WString	Character: Variable Length Wide String	If the string greater than 16 characters and varies in length from value to value. If the string contains unicode and is longer than
Date	Character: A 10 character String in "yyyy-mm-dd" format	December 2, 2005 = 2005-12-02
Time	Character: A 8 character String in "hh:mm:ss" format	2:47 and 53 seconds, pm = 14:47:53
DateTime	Character: A 19 character String in "yyyy-mm-dd hh:mm:ss" format	2005-12-02 14:47:53
Blob	Blob: Binary Large Object: A large block of data stored in a database. A BLOB has no structure which can be interpreted by the database management system but is known only by its size and location.	an image or sound file
SpatialObj	Blob: The spatial object associated with a data record. There can be multiple spatial object fields contained within a table.	A spatial object can consist of a point, line, polyline, or polygon.

Flat files are intended to be used with ASCII characters.

Quick Reference For All Tools

https://help.alteryx.com/10.6/Getting_Started/AllTools.htm

Browse Colors

View data in the Configuration window and the Results window. The Configuration window displays different charts

A colored data quality bar displays at the top of each column of data in the Results window.

- Red (Not OK): The column contains values with leading or trailing white space.
- Yellow (Null): The column contains no values.
- Gray (Empty): The column contains strings with no values.
- Green (OK): The column contains values without leading or trailing white spaces.

Browse Metadata

Numeric data

If the selected column contains numeric values, the following metadata is provided:

- **Name**: The column name.
- **Data Type**: The data type of the selected column.
- **Size**: The amount of memory reserved for each record in this column.
- **Non-Nulls**: The number of non-null entries in the column, including empty values.
- **Uniques**: The number of unique values in the field. Use the Unique tool to see a full count of unique and duplicate entries. See Unique Tool.
- **Nulls**: The number of values in the column that are null, excluding empty values.
- **Minimum**: The smallest value in the column.
- **Maximum**: The largest value in the column.
- **Average**: The average value of values in the column.
- **Standard Deviation**: The measure of how dispersed the values are in the chart.
- **Variance**: The measure of how far a set of random numbers are dispersed from the mean.
- **25th Percentile**: The median value in the lower, or first, half of the data.
- **50th Percentile**: The median value of the data.
- **75th Percentile**: The median value in the upper, or second, half of the data.

String data

If the selected column contains string values, the following metadata is provided:

- **Name**: The column name.
- **Data Type**: The data type of the selected column.
- **Size**: The amount of memory reserved for each record in this column.
- **Non-Nulls**: The number of non-null entries in the column, including empty values.
- **Uniques**: The number of unique values in the field. Use the Unique tool to see a full count of unique and duplicate entries. See Unique Tool.
- **Nulls**: The number of values in the column that are null, excluding empty values.
- **Blanks**: The number of empty values.
- **Values with Leading Whitespace**: The number of string values with whitespace before the value. Use the Data Cleansing tool or the Formula
- **Values with Trailing Whitespace**: The number of string values with whitespace after the value.
- **Shortest (Non-Blank) Length**: The number of characters in the shortest value in the column.
- **Average Length**: The average length of values in the column.
- **Longest Length**: The number of characters in the longest value in the column.
- **Shortest Value**: The shortest value in the column.
- **Longest Value**: The longest value in the column.
- **First Alphanumeric Value**: The first string entry in a column that is sorted alphabetically.
- **Last Alphanumeric Value**: The last string entry in a column that is sorted alphabetically.

Date/Time data

If the selected column contains date/time values, the following metadata is provided:

- **Name**: The column name.
- **Data Type**: The data type of the selected column.
- **Size**: The amount of memory reserved for each record in this column.
- **Non-Nulls**: The number of non-null entries in the column, including empty values.
- **Uniques**: The number of unique values in the field. Use the Unique tool to see a full count of unique and duplicate entries. See Unique Tool.
- **Nulls**: The number of values in the column that are null, excluding empty values.
- **Minimum**: The smallest value in the column.
- **Maximum**: The largest value in the column.

Spatial data

If the selected column contains spatial objects, the following metadata is provided:

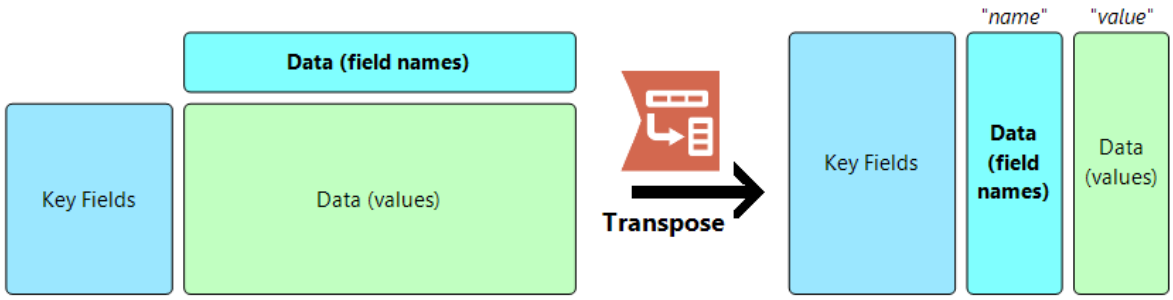
- **Name**: The column name.
- **Data Type**: The data type of the selected column.
- **Size**: The amount of memory reserved for each record in this column.
- **Non-Nulls**: The number of non-null entries in the column, including empty values.
- **Nulls**: The number of values in the column that are null, excluding empty values.

Overview

The Transpose tool pivots the data so that a wide data set becomes a narrower data set.

All of the fields selected as Data Fields will have their values stacked into a single column called "Value", while the names of each corresponding variable will be stored alongside in a column called "Name". The Key Fields will remain unchanged, and all other columns will be dropped from the output.

Below is a conceptual representation of how the Transpose tool transforms the data.

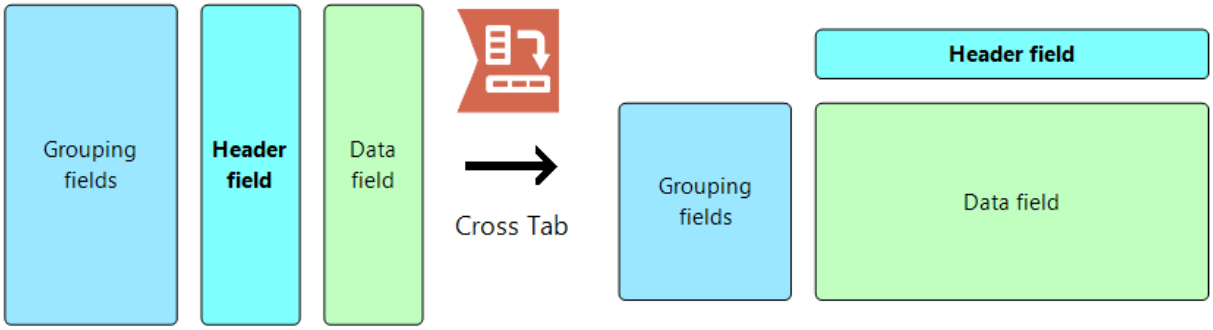


Overview

The Cross Tab tool pivots the data so that a narrow data set becomes a wider data set.

A new column is created for each unique value in the New Column Headers field, which will be populated by aggregating the values in the Values for New Columns field, while the contents of the Group Data by these Values field do not change.

Below is a conceptual representation of how the Cross Tab tool transforms the data.

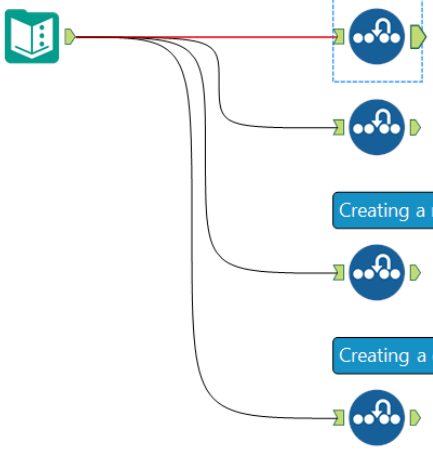


Imputation Tool

The Imputation tool replaces problematic numeric values (such as nulls) for specified columns with another value such as the median or a user-specified value.

1) Run the workflow (Ctrl+R).

2) Select a tool to view its output in the Results window.



Replacing null values

The most common use of the Imputation tool is to replace null values with another number. This can be the average, median, mode of the column, or a user-specified value such as zero. The user can also specify a value other than [Null] to be replaced if desired.

The first example has null values being replaced with zero, while the second example has null values being replaced with the average of the column.

Creating a new column with imputed values

The "Output imputed values as separate field" option leaves the original fields as they were as creates new ones with the suffix "_ImputedValue".

Creating a column indicating whether or not a value has been imputed

The "Include imputed value indicator field" creates a binary column for each field selected with the suffix "_Indicator", where 1 indicates that the value has been imputed and 0 indicates that it has been left unchanged.

More Info

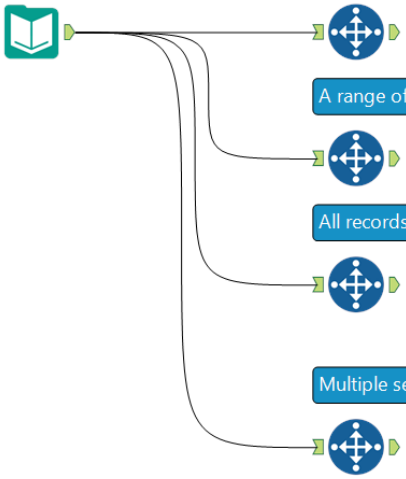
It is often a good idea to use the Imputation before streaming data into the predictive tools since many are designed to error when null values are encountered. However, be careful to choose an appropriate replacement value to get the best results from the predictive tools.

Select Records Tool

The Select Records tool can be used to choose a precise subset of input records.

1) Run the workflow (Ctrl+R).

2) Select a tool to view its output in the Results window.



A single record

The Select Records tool can be used to specify a single record, such as 10800 in this example.

A range of records

A hyphen is used to specify a range, in this case rows 109-113.

All records beyond a certain row

A plus (+) is used to select records starting at 20000 and also return all rows through the last record in the set.

Alternately, a minus (-) can be used to return all records preceeding a certain record. (see below)

Multiple selections

Multiple selections can listed together.

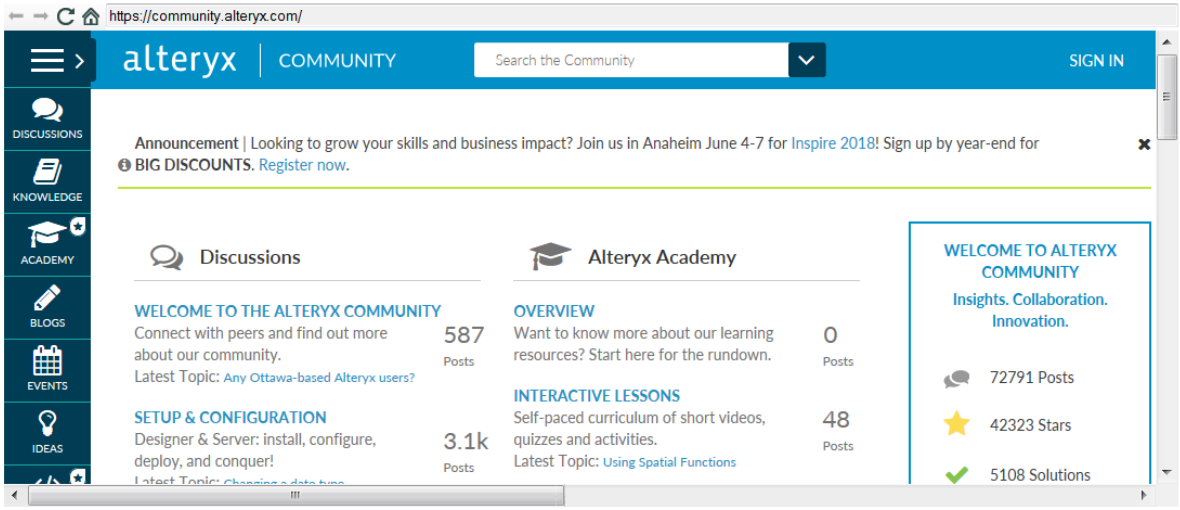
This Select Records tool is configured to return rows 1-87, 109-113, 10800, and 20000 through the end of the data set.

Explorer Box Tool

Display a web page, file directory, or file on the canvas.

Display a web page

Enter a URL to display a web page when an internet connection is available. To resize the box, click it and then click and drag the squares on any side to the desired position.



Display file directory

Add a directory path to display a file directory. Double-click a file in the directory to open it in the default program for that file type, or click and drag data, workflow, analytic app, or macro files to the canvas.

Supported Data Sources

Alteryx connects to a variety of data sources. Alteryx can read, write, or read and write, dependent upon the data source.

Adobe	Adobe Analytics		Read-only	
Alteryx	Alteryx Database	.yxdb	Read & Write	
	Alteryx Calgary	.cydb	Read	
	Alteryx Spatial Zip	.sz	Read & Write	
Amazon	Amazon Aurora		Read & Write	
	Amazon Redshift		Read & Write	In-DB Support
	Amazon S3		Read & Write	
ASCII	ASCII	.flat, .asc	Read & Write	
Apache Hadoop	Apache Hadoop Avro	.avro	Read & Write	
	Cassandra		Read & Write	
	Hadoop Distributed File System (HDFS)		Read & Write	
	Hive		Read & Write	In-DB Support
	Spark		Read & Write	In-DB Support
Autodesk	Autodesk	.sdf	Read & Write	
Cloudera	Cloudera Impala		Read & Write	In-DB Support
	Hadoop Distributed File System (HDFS)		Read & Write	
	Hive		Read & Write	In-DB Support
CSV	Comma Separated Value	.csv	Read & Write	
Databricks	Databricks		Read & Write	In-DB Support
DataStax	DataStax Enterprise, DataStax Community		Read & Write	
dBase	dBase	.dbf	Read & Write	
ESRI	ESRI GeoDatabase	.gdb	Read-only	
	ESRI Personal GeoDatabase	.mdb	Read-only	
	ESRI Shapefile	.shp (.dbf, .shx, .prj)	Read & Write	
EXASOL	EXASOL		Read & Write	In-DB Support
Foursquare	Foursquare		Read-only	
GIS	GIS	.grd, .grc	Read-only	
Google	Google Analytics		Read-only	
	Google BigQuery		Read-only	
	Google Earth/Maps	.kml	Read & Write	
	Google Sheets		Read & Write	
Hortonworks	Hadoop Distributed File System (HDFS)		Read & Write	
	Hive		Read & Write	In-DB Support
HP	Vertica		Read & Write	In-DB Support
HTML	HyperText Markup Language	.htm	Write	
IBM	IBM DB2		Read & Write	
	IBM Netezza/Pure Data Systems		Read & Write	In-DB Support
	IBM SPSS	.sav	Read & Write	
JSON	JSON	.json	Read & Write	
MapInfo	MapInfo Professional Interchange Format	.mid, .mif	Read & Write	
	MapInfo Professional Table	.tab (*.dat, *.map, *.id, *.ind)	Read & Write	

MapR	Hadoop Distributed File System (HDFS)		Read & Write	
	Hive		Read & Write	In-DB Support
Marketo	Marketo		Read & Write	
Microsoft	Microsoft Access 2000-2003	.mdb	Read & Write	
	Microsoft Analytics Platform System		Read & Write	In-DB Support
	Microsoft Azure ML		Read-only	
	Microsoft Azure SQL Database		Read & Write	In-DB Support
	Microsoft Azure SQL Data Warehouse		Read & Write	In-DB Support
	Microsoft Cognitive Services		Read-only	
	Microsoft Excel 1997-2003	.xls	Read & Write	
	Microsoft Excel 2007, 2010, 2013, 2016	.xlsx	Read & Write	
	Microsoft Excel Macro Enabled	.xlsm	Read & Write	
	Microsoft Office Access 2007, 2010, 2013, 2016 Requires MS Office Access or driver	.accdb	Read & Write	
	Microsoft Power BI		Write	
	Microsoft SQL Server 2008, 2012, 2014, 2016		Read & Write	In-DB Support
	Microsoft SharePoint		Read & Write	
MongoDB	MongoDB		Read & Write	
MySQL	MySQL		Read & Write	
Netsuite	Netsuite Suite Analytics		Read-only	
OpenGIS	Geography Markup Language	.gml	Read & Write	
Oracle	Oracle		Read & Write	In-DB Support & Predictive Support
Pivotal	Pivotal Greenplum		Read & Write	
PostgreSQL	PostgreSQL		Read & Write	
Qlik	Qlik Sense, QlikView	.qvx	Read & Write	
Salesforce.com	Salesforce		Read & Write	
	Salesforce Wave			
SAP	SAP Hana		Read & Write	In-DB Support
	Sybase Adaptive Server Enterprise		Read & Write	
	Sybase SQL Anywhere 10		Read & Write	
SAS	SAS	.sas7bdat	Read & Write	
SQLite	SQLite	.sqlite	Read & Write	
SRC Geography	SRC Geography File	.geo	Read & Write	
Tableau	Alteryx Web Data Connector for Tableau		Publish	
	Publish to Tableau Server		Publish	
	Tableau Data Extract	.tde	Write-only	
Teradata	Teradata		Read & Write	In-DB Support
	Teradata Aster		Read	
Text	Text	.txt	Read	
Twitter	Twitter		Read-only	
XML	Extensible Markup Language	.xml	Read-only	
Zip Files	Zip Files	.zip	Read-only	

Data Types

String data

A string represents alphanumeric data and can include letters, numbers, spaces, or other types of characters. A string can also be thought of as plain text. All the characters in a string are considered text even if the characters are digits.

> While a string may contain text that looks like a number (for example, "123.4"), it must first be converted to a numeric data type (either with a Select Tool, or with the ToNumber Functions) to perform calculations.

Type	Description	Example
String	Fixed Length Latin-1 String. The length should be at least as large as the longest string you want contained in the field or values will be truncated. Limited to 8192 Latin-1 characters.	Any string whose length does not vary much from value to value, and only contains simple Latin-1 characters.
WString	Wide String will accept any character (Unicode). Limited to 8192 characters.	Any string whose length does not vary much from value to value and contains any character.
V_String	Variable Length. The length of the field will adjust to accommodate the entire string within the field.	Any string whose length varies from value to value, and only contains simple Latin-1 characters.
V_WString	Variable Length Wide String. The length of the field will adjust to accommodate the entire string within the field and will accept any character.	Any string whose length varies from value to value and contains any character.

Numeric data

There are several different numeric data types including integers, decimals, floats, and doubles. Numeric data types do not have adjustable lengths except for Fixed Decimal.

Type	Description	Example
Byte	A unit of data that is 8 binary digits (bits) long. A byte field is a positive whole number that falls within the range 0 thru 255, or 2^8	0, 1, 2, 3....253, 254, 255
Int16	A numeric value without a decimal equal to 2 bytes, or $-(2^{15})$ to $(2^{15})-1$	−32,768 to 32,767
Int32	A numeric value without a decimal equal to 4 bytes, or $-(2^{31})$ to $(2^{31})-1$	−2,147,483,648 to 2,147,483,647
Int64	A numeric value without a decimal equal to 8 bytes, or $-(2^{63})$ to $(2^{63})-1$	−9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
Fixed Decimal	A numeric value with a decimal. The length (precision) of a fixed decimal is equal to the width of the integer (left side of decimal) plus the decimal point plus the width of the scale (right side of decimal). If a number is negative, the negative sign is also included in the length. Alteryx defaults a Fixed Decimal to 19.6. The maximum precision is 50, inclusive of the decimal point and negative sign (if applicable). A Fixed Decimal is the only numeric data type with an adjustable length.	A value of 1234.567 with a length of 7.2 results in 1234.57 A value of 1234.567 with a length of 7.3 results in a field conversion error and Null output, as the value does not fit within the specified precision. A value of 1234.567 with a length of 6.1 results in 1234.6 A value of 1234.567 with a length of 8.3 results in 1234.567 A value of -1234.567 with a length of 8.3 results in a field conversion error and Null output, as the value does not fit within the specified precision. A value of 1234.567 with a length of 11.6 results in 1234.567000
Float	A standard single precision floating point value. It uses 4 bytes, and can represent values from $+/- 3.4 \times 10^{-38}$ to 3.4×10^{38} with 7 digits of precision. A float uses a decimal that can be placed in any position and is mainly used to save memory in large arrays of floating point numbers.	$+/- 3.4 \times 10^{-38}$ to 3.4×10^{38} with 7 digits precision
Double	A standard double precision floating point value. It uses 8 bytes, and can represent values from $+/- 1.7 \times 10^{-308}$ to 1.7×10^{308} with 15 digits precision. A double uses a decimal that can be placed in any position. A double uses twice as many bits as a float and is generally used as the default data type for decimal values.	$+/- 1.7 \times 10^{-308}$ to 1.7×10^{308} with 15 digits precision

Date/Time data

Type	Description	Example
Date	A 10 character String in "yyyy-mm-dd" format	December 2, 2005 = 2005-12-02
Time	A 8 character String in "hh:mm:ss" format	2:47 and 53 seconds a.m. = 02:47:53 2:47 and 53 seconds p.m. = 14:47:53
DateTime	A 19 character String in "yyyy-mm-dd hh:mm:ss" format	2005-12-02 14:47:53

Boolean data

Type	Description	Example
Bool	An expression with only two possible values: True or False	The words 'True' and 'False' display in the results where 'False' = 0 and 'True' = non-zero.

Spatial objects

Type	Description	Example
SpatialObj	The spatial object associated with a data record. There can be multiple spatial object fields contained within a table.	A spatial object can consist of a point, line, polyline, or polygon.

Boost Regex

Boost-Extended Format String Syntax

Boost-Extended format strings treat all characters as literals except for '$', '\', '(', ')', '?', and ':'.

Grouping

The characters '(' and ')' perform lexical grouping, so use \(and \) if you want a to output literal parenthesis.

Conditionals

The character '?' begins a conditional expression, the general form is:

?Ntrue-expression:false-expression

where N is decimal digit.

If sub-expression N was matched, then true-expression is evaluated and sent to output, otherwise false-expression is evaluated and sent to output.

You will normally need to surround a conditional-expression with parenthesis in order to prevent ambiguities.

For example, the format string "(?1foo:bar)" will replace each match found with "foo" if the sub-expression $1 was matched, and with "bar" otherwise.

For sub-expressions with an index greater than 9, or for access to named sub-expressions use:

?{INDEX}true-expression:false-expression

or

?{NAME}true-expression:false-expression

Placeholder Sequences

Placeholder sequences specify that some part of what matched the regular expression should be sent to output as follows:

Placeholder	Meaning
$&	Outputs what matched the whole expression.
$MATCH	As $&
${^MATCH}	As $&
$`	Outputs the text between the end of the last match found (or the start of the text if no previous match was found), and the start of the current match.
$PREMATCH	As $`
${^PREMATCH}	As $`
$'	Outputs all the text following the end of the current match.
$POSTMATCH	As $'
${^POSTMATCH}	As $'
$+	Outputs what matched the last marked sub-expression in the regular expression.
$LAST_PAREN_MATCH	As $+
$LAST_SUBMATCH_RESULT	Outputs what matched the last sub-expression to be actually matched.
$^N	As $LAST_SUBMATCH_RESULT
$$	Outputs a literal '$'
$n	Outputs what matched the n'th sub-expression.
${n}	Outputs what matched the n'th sub-expression.
$+{NAME}	Outputs whatever matched the sub-expression named "NAME".

Escape Sequences

An escape character followed by any character x, outputs that character unless x is one of the escape sequences shown below.

Escape	Meaning
\a	Outputs the bell character: '\a'.
\e	Outputs the ANSI escape character (code point 27).
\f	Outputs a form feed character: '\f'
\n	Outputs a newline character: '\n'.
\r	Outputs a carriage return character: '\r'.
\t	Outputs a tab character: '\t'.
\v	Outputs a vertical tab character: '\v'.
\xDD	Outputs the character whose hexadecimal code point is 0xDD
\x{DDDD}	Outputs the character whose hexadecimal code point is 0xDDDDD
\cX	Outputs the ANSI escape sequence "escape-X".
\D	If D is a decimal digit in the range 1-9, then outputs the text that matched sub-expression D.
\l	Causes the next character to be outputted, to be output in lower case.
\u	Causes the next character to be outputted, to be output in upper case.
\L	Causes all subsequent characters to be output in lower case, until a \E is found.
\U	Causes all subsequent characters to be output in upper case, until a \E is found.
\E	Terminates a \L or \U sequence.

Alteryx Keyboard Shortcuts

Shortcuts

This page describes keyboard, mouse, and right-click context menu shortcuts in Alteryx. You can use these shortcuts to perform many funct

Select and Align Tools

Action	Shortcut
Select all items	Ctrl + A
Deselect all selected items	Ctrl + D
Align tools vertically	Ctrl + Shift + +
Align tools horizontally	Ctrl + Shift + -

Move and Delete Tools

Action	Shortcut
Move selected tool	Arrow key
Move selected tool by one pixel	Ctrl + Arrow key
Delete selected tool	Delete

Scroll and Pan the Canvas

Action	Shortcut
Scroll vertically	Scroll Function
Scroll horizontally	Shift + Scroll Function
Move up, down, left or right	Arrow Keys
Skip up, down, left or right	Shift + Arrow Keys
Jump to top or bottom	Home or End
Jump to left or right	Shift + Home or End
Pan	Space Bar + Left Click
Pan	Hold mouse center button

Zoom In and Out of the Workflow

Action	Shortcut
Zoom in or out	Ctrl + + or -
Zoom to or from cursor	Ctrl + Scroll Function
Zoom to entire workflow	Ctrl + 0
Zoom to entire workflow	Double-click mouse center button
Jump to selection	Ctrl + 0
Jump in or out	Ctrl + 1 through 5
Zoom to area	Right-click & drag to select with magnifying glass

Zoom In and Out of the Workflow: Right-Click Context Menu

Action	Right-click Context Menu
Zoom in	Zoom > Zoom In
Zoom normal	Zoom > Zoom Normal
Zoom out	Zoom > Zoom Out
Zoom to entire workflow	Zoom > All
Zoom to selection	Zoom > Selected Tools
Zoom to container	Zoom > Container Name

Show and Hide Tools and Windows

Action	Shortcut
Show Toolbar	Ctrl + Alt + B
Show Tool Palette	Ctrl + Alt + T
Show Overview	Ctrl + Alt + V
Show Results Window	Ctrl + Alt + R
Show Configuration Window	Ctrl + Alt + C
Reopen the Configuration Window after closing it	Double-click the canvas or any tool
Show the Interface Designer	Ctrl + Alt + D
Show the Find Tool window	Ctrl + F

Run, Open, Save, and Switch Workflows

Action	Shortcut
Run workflow; stop workflow from running	Ctrl + R
Open workflow	Ctrl + O
Close active workflow	Ctrl + F4
Save workflow	Ctrl + S
New workflow	Ctrl + N
Move between active workflows	Ctrl + Tab

Undo and Redo, Copy and Paste

Action	Shortcut
Undo	Ctrl + Z
Redo	Ctrl + Y
Copy	Ctrl + C
Cut	Ctrl + X
Paste	Ctrl + V

Other Shortcuts

Action	Shortcut
Open help page for the selected tool	F1
Close Alteryx Designer	Alt + F4
Add a Browse tool after selected tool(s). If the selected tool has multiple outputs, a browse will be added for each.	Ctrl + Shift + B
Activate menu and select menu item	Alt, then *underlined menu letter*
Show the System menu	Alt + Space
Open one or more files: • Open a workflow file (*.yxmd, *.yxwz, *.yxmc) directly in a new tab on the workflow canvas. • Open a data file as a configured Input Tool.	Drag files from Windows Explorer directly to the canvas.
Copy and paste a tool's color value to another tool. For example, if you have selected a tool's Background Color (such as R=73, G=248, B=113) and you would like to use this same color for another tool's background color, you can easily copy and paste this color value.	1. Click inside the box containing the color values, and click Ctrl+C to copy the color values. 2. Click inside another color box, and click Ctrl+V to paste the color values.
Refresh Workflow Use F5 to refresh tool configurations when the incoming data source has been updated externally and the metadata has changed. If the Disable Auto Configure option has been selected in the User Settings, press F5 to manually refresh tool configurations.	F5