Expressions – Aggregations on the Fly

Release Date

2017 Q4

Summary

Expressions can contain multiple aggregated values that can be used in the same way as variables and the resulting expressions can be used throughout Market Insight in places where expressions can be used.

In the past the use of aggregated data in an expression had to be created through wizards to generate an appropriate virtual variable. It is now possible to create aggregations 'on the fly' within the expression tool. Multiple aggregations can be created in a single expression and combined together as if they were Market Insight variables. These expressions can then be used in all the places you would expect to be able to use expressions through Market Insight.

The main aggregation types supported are the same as in the RFV function:

Recency – e.g. Last(Policy) a Customer has purchased
Frequency – e.g. Number of Policies for a Record
Value – e.g. Mean(Cost) of Policies for a Record

The following further aggregations have been added where the result is on the transactional table. These new functions allow analysis questions to be answered that have not been possible before.

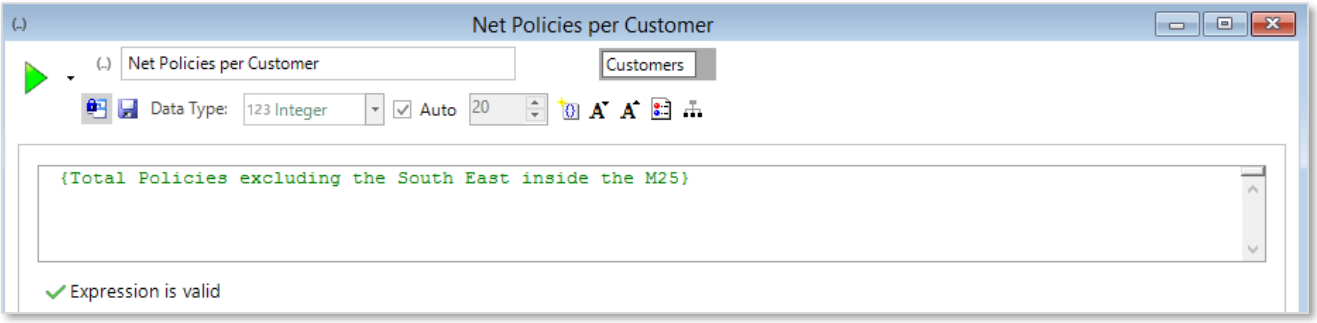
Rank Transaction – e.g. Rank a policy for a record
Relative Transaction – e.g. The cost of the next policy taken

In all of these cases the transactional table can have a filter selection defined to choose only a subset of transactional records.

The following examples will demonstrate each of these aggregations within an expression.

Frequency Aggregation Expression Example

Scenario: What are the net number of Policies per customer once Policies taken by Customers inside the M25 have been removed?



Net Policies per Customer

Net Policies per Customer | Customers

Data Type: 123 Integer ☑ Auto 20

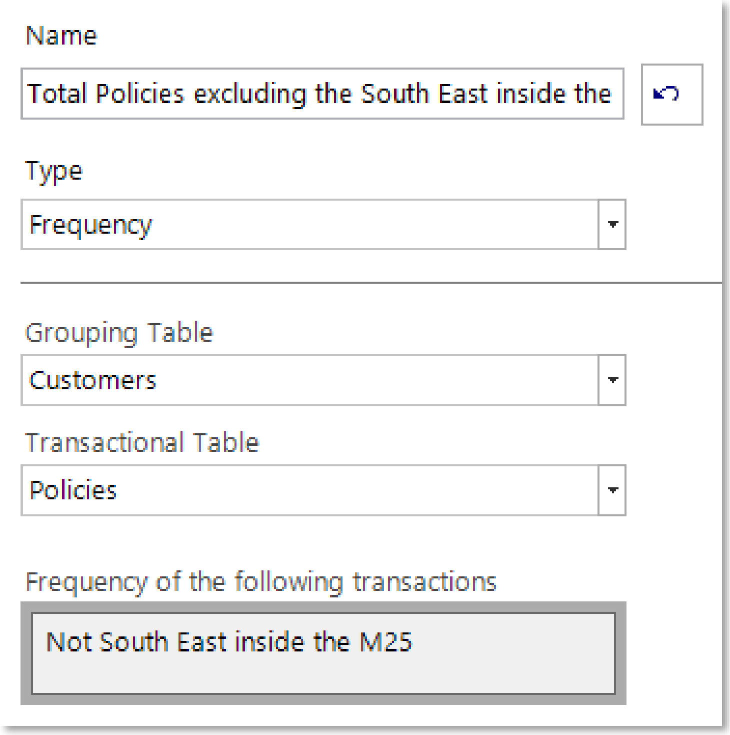
```
{Total Policies excluding the South East inside the M25}
```

✓ Expression is valid

- ➢ Click on the Add Aggregation button and ensure the table level is **Customers** (this will be the table level of all aggregations in this document unless otherwise stated)
- ➢ Click on the tab entitled **Frequency(Policies)**
- ➢ Create a selection that excludes **Policies within the M25** and drag it onto the filter box

N.B. The auto naming feature will populate the Name box until changed by the User.

- ➢ Change the **Name** to display **Total Policies excluding the South East inside the M25**
- ➢ Click the **Build** button to see a preview of the results
- ➢ Use a **Data Grid** to validate the results (Variables: Your Expression, Policy Product Type)



Name: Total Policies excluding the South East inside the

Type: Frequency

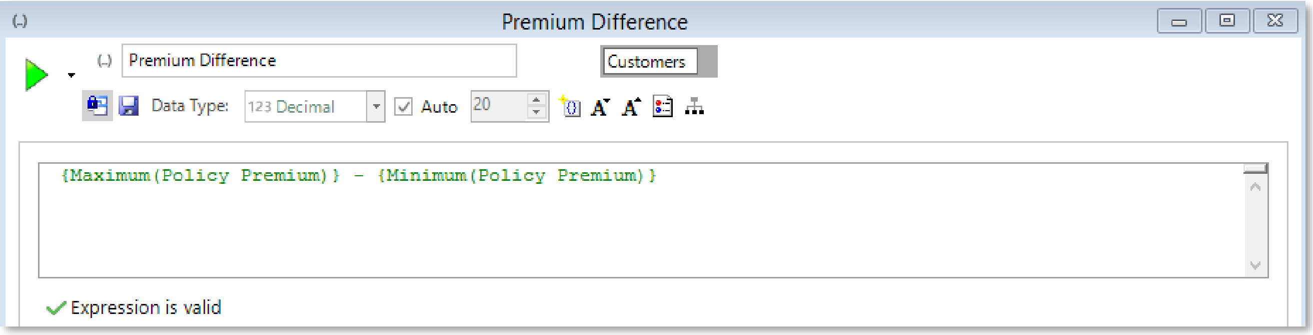
Grouping Table: Customers

Transactional Table: Policies

Frequency of the following transactions: Not South East inside the M25

Value Aggregation Expression Example

Scenario: What is the difference between each person’s maximum and minimum Policy Premium



Premium Difference

Premium Difference

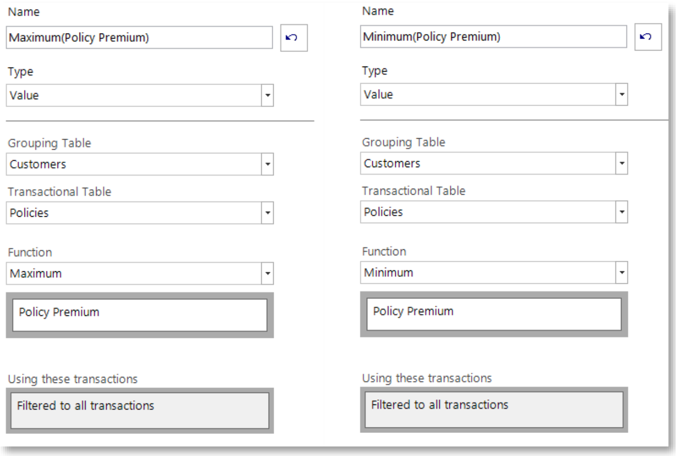
Customers

Data Type: 123 Decimal ☑ Auto 20

```
{Maximum(Policy Premium)} - {Minimum(Policy Premium)}
```

✓ Expression is valid

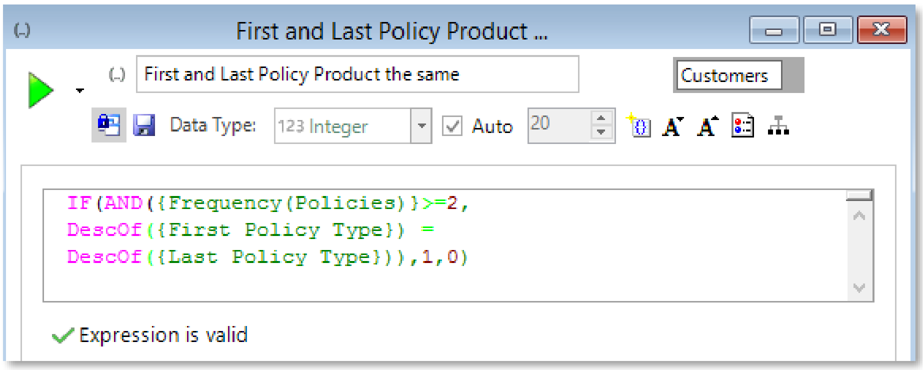
- Click on the **Add Aggregation** button
- Click on the tab entitled **Frequency(Policies)**
- Change the **Type** box to show **Value**
- Drag the **Policy Premium** variable onto the **Function** drop box and change the function to display **Maximum**
- Click on the **Expression** tab and enter a **minus** sign after the aggregation title
- Click on the **Add Aggregation** button to generate a new aggregation tab
- Click on the tab entitled **Frequency(Policies)**
- Change the **Type** box to show **Value**
- Drag the **Policy Premium** variable onto the **Function** drop box and change the function to display **Minimum**
- Click the **Build** button to see a preview of the results
- Use a **Data Grid** to validate the results (Variables: Your Expression, Policy Premium)



Name: Maximum(Policy Premium)	Name: Minimum(Policy Premium)
Type: Value	Type: Value
Grouping Table: Customers	Grouping Table: Customers
Transactional Table: Policies	Transactional Table: Policies
Function: Maximum	Function: Minimum
Policy Premium	Policy Premium
Using these transactions: Filtered to all transactions	Using these transactions: Filtered to all transactions

Recency Aggregation Expression Example

Scenario: Identify Customers who have more than 2 Polices where the Product Type of their first and last Policies are the same.



First and Last Policy Product ...

(..) First and Last Policy Product the same

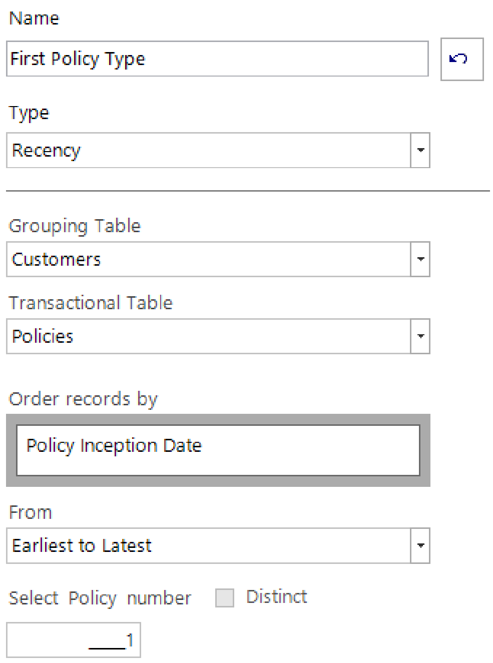
Customers

Data Type: 123 Integer ☑ Auto 20

```
IF(AND({Frequency(Policies)}>=2,
DescOf({First Policy Type}) =
DescOf({Last Policy Type})),1,0)
```

✓ Expression is valid

- Enter an **If(** function followed by an **And(** function
- Click on the **Add Aggregation** button and then enter **>2** after the {Frequency(Policies)} statement followed by a **comma**
- Enter the **DescOf(** function and then click on the **Add Aggregation** button
- Click on the **2.Frequency(Policies)** tab and change the **Type** box to **Recency**
- Drag the **Policy Inception Date** variable on to the **Order records by** drop box
- Drag the **Policy Product Type** variable onto the **Pick the** drop box
- Change the **Name** to **First Policy Type**
- In the expression window enter a closed bracket **)** followed by the equals symbol **=**
- Enter the **DescOf(** function and then click on the **Add Aggregation** button
- Click on the new **2.Frequency(Policies)** tab and change the **Type** box to **Recency**
- Drag the **Policy Inception Date** variable on to the **Order records by** drop box
- Change the **From** box to show **Latest to Earliest**
- Drag the **Policy Product Type** variable onto the **Pick the** drop box
- Change the **Name** to **Last Policy Type**
- In the expression window enter the following after {Last Policy Type} **)),1,0)**
- Click the **Build** button to see a preview of the results
- Use a **Data Grid** to validate the results (Variables: Your Expression, Policy Product Type, Policy Inception Date)



Name

First Policy Type

Type

Recency

Grouping Table

Customers

Transactional Table

Policies

Order records by

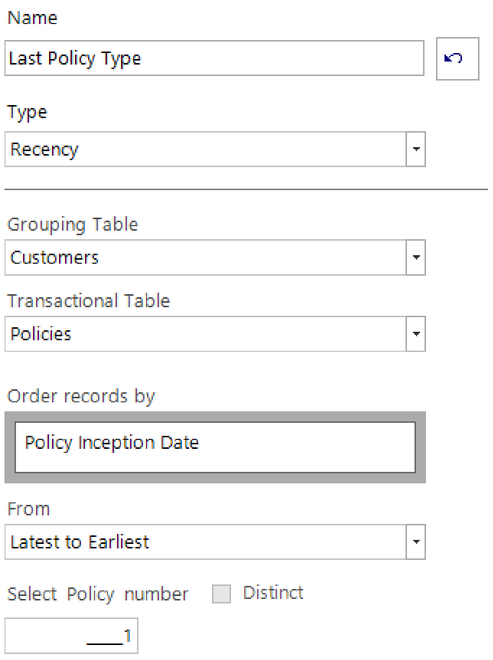
Policy Inception Date

From

Earliest to Latest

Select Policy number □ Distinct

1



Name

Last Policy Type

Type

Recency

Grouping Table

Customers

Transactional Table

Policies

Order records by

Policy Inception Date

From

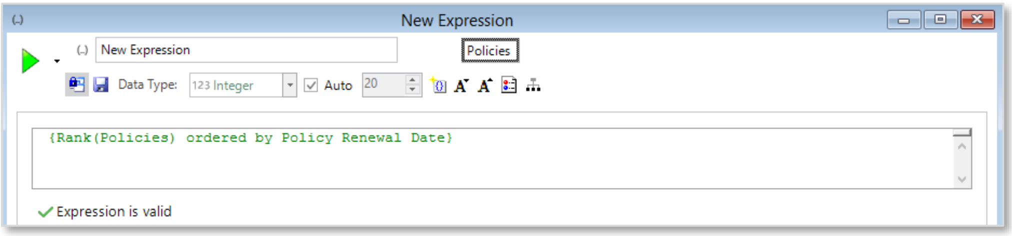
Latest to Earliest

Select Policy number □ Distinct

1

Rank Transaction Aggregation Expression Example

Scenario: Rank a customer's policies by Policy Renewal Date(Earliest to Latest)



New Expression

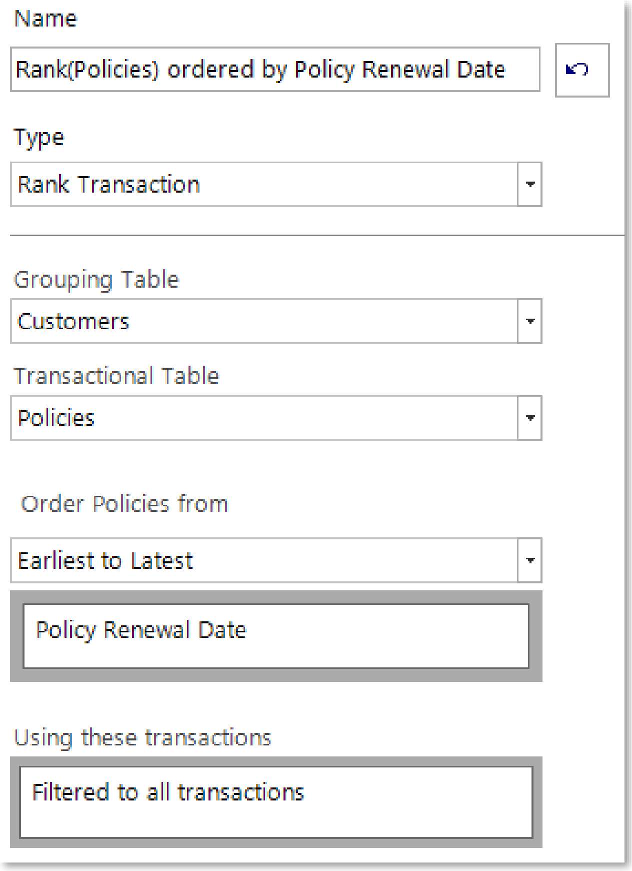
Policies

Data Type: 123 Integer ☑ Auto 20

```
{Rank(Policies) ordered by Policy Renewal Date}
```

✓ Expression is valid

- Click on the **Add Aggregation** button
- Click on the tab entitled **Frequency(Policies)**
- Change the **Type** box to show **Rank Transaction**
- Drag the **Policy Renewal Date** variable onto the **Ordering** drop box and leave the order display as **Earliest to Latest**
- Click on the Expression tab and change the table level to **Policies**
- Click the **Build** button to see a preview of the results
- Use a **Data Grid** to validate the results (Variables: Your Expression, Policy Renewal Date)



Name

Rank(Policies) ordered by Policy Renewal Date

Type

Rank Transaction

Grouping Table

Customers

Transactional Table

Policies

Order Policies from

Earliest to Latest

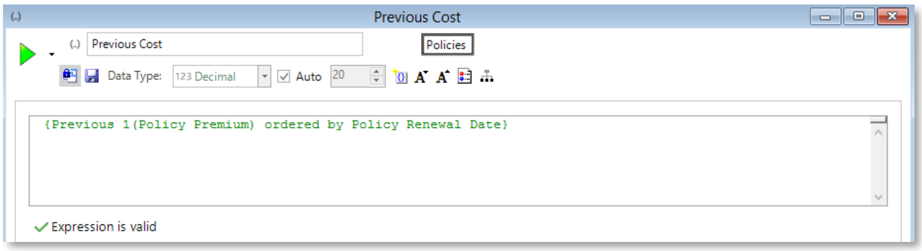
Policy Renewal Date

Using these transactions

Filtered to all transactions

Relative Transaction Aggregation Expression Example

Scenario: When all a customer's policies have been ranked by Policy Renewal Date find the Premium of the previous Policy.



(-) Previous Cost

Policies

Data Type: 123 Decimal ☑ Auto 20

```
{Previous 1(Policy Premium) ordered by Policy Renewal Date}
```

✓ Expression is valid

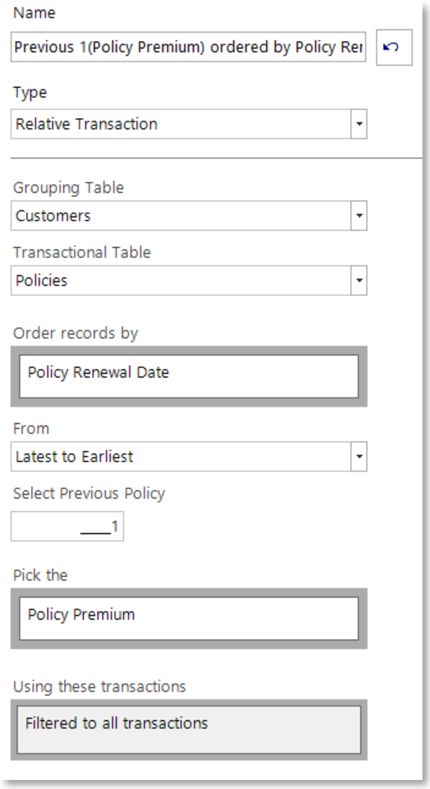
- Click on the 品 **Add Aggregation** button
- Click on the tab entitled **Frequency(Policies)**
- Change the **Type** box to show **Relative Transaction**
- Drag the **Policy Renewal Date** variable onto the **Ordering** drop box and change the order display to **Latest to Earliest**
- Leave the **Select Previous Policy** set to **1** to find the previous order, as opposed to 2 to find the second previous order etc.

- Drag the **Policy Premium** variable on to the **Pick the** drop box

- Click on the **Expression** tab and change the table level to **Policies**

- Click the ▶ **Build** button to see a preview of the results

- Use a **Data Grid** to validate the results (Variables: Your Expression, Policy Renewal Date, Policy Premium)



Name
Previous 1(Policy Premium) ordered by Policy Re

Type
Relative Transaction

Grouping Table
Customers

Transactional Table
Policies

Order records by
Policy Renewal Date

From
Latest to Earliest

Select Previous Policy
1

Pick the
Policy Premium

Using these transactions
Filtered to all transactions

N.B. This type of function can be used to look for increased costs between transactions or patterns in the types of products bought etc.