AccumulateBy

Iterates through the records in the pipe, groups them based on the provided extractor, and then reduces each group to a single record per group with the accumulator function.

The initial value of the accumulator is null unless you provide a value initializer operation as a parameter.

Parameters

Type parameters:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>The template type of the returned builder</td>
</tr>
</tbody>
</table>

Function parameters:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accumulator</td>
<td>AccumulateByOperation&lt;T,I&gt;</td>
<td>A function with logic to update the accumulator value with each record</td>
</tr>
<tr>
<td>extractor</td>
<td>ExtractorOperation</td>
<td>Extracts a specific value from each record</td>
</tr>
<tr>
<td>valueInitializer</td>
<td>ValueInitializerOperation</td>
<td>Whenever the accumulated value is null, use this function to initialize it</td>
</tr>
</tbody>
</table>

Returns

Returns a GearsBuilder object with a new template type.

Examples

Both of the following examples count the number of unique values.

Without the valueInitializer parameter:
GearsBuilder.CreateGearsBuilder(reader).
   accumulateBy(r->{
      return r.getStringVal();
   },(k, a, r)->{
      Integer ret = null;
      if(a == null) {
         ret = 0;
      } else {
         ret = (Integer)a;
      }
      return ret + 1;
   });

With the valueInitializer parameter:

GearsBuilder.CreateGearsBuilder(reader).
   accumulateBy(()->{
      return 0;
   },r->{
      return r.getStringVal();
   },(k, a, r)->{
      return a + 1;
   });

Updated: January 31, 2022