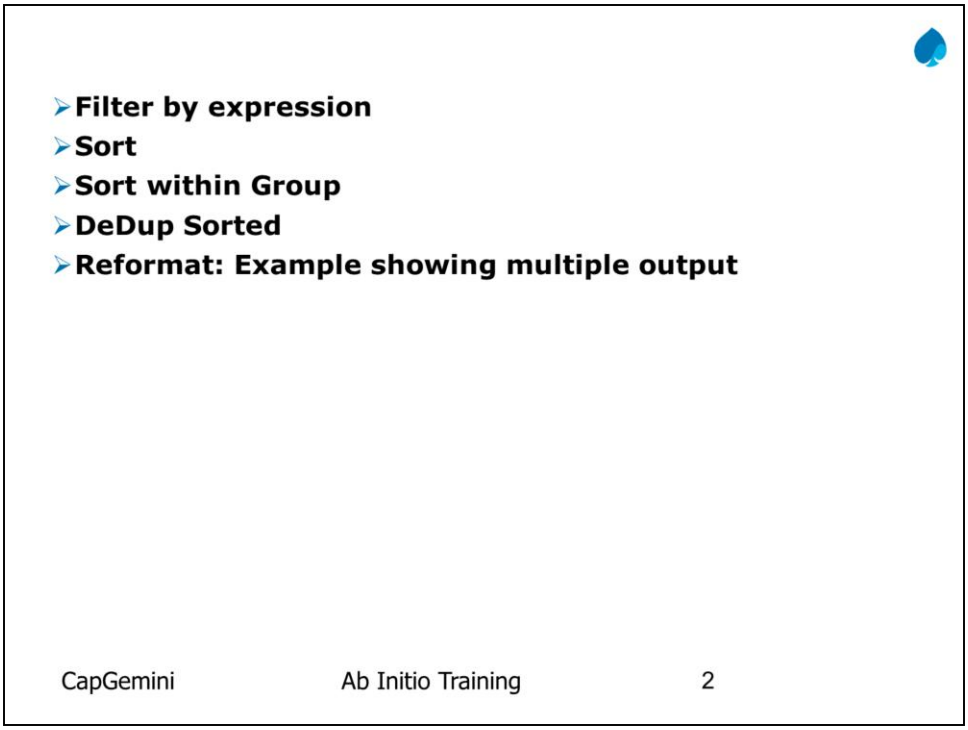

Ab initio Session 3
Ab Initio Components
Capgemini
Ab Initio Training
1


Filter by expression
Sort
Sort within Group
DeDup Sorted
Reformat: Example showing multiple output
CapGemini
Ab Initio Training
2

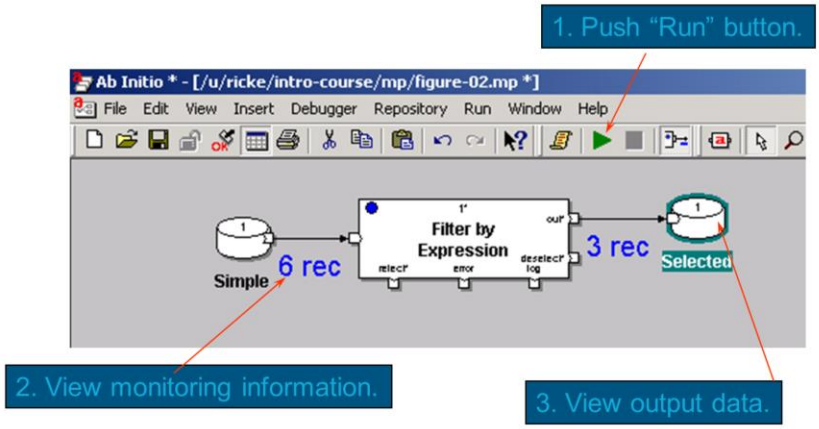
Filter by Expression

- Filter by Expression filters data records according to a DML expression.

The Filter by Expression Component

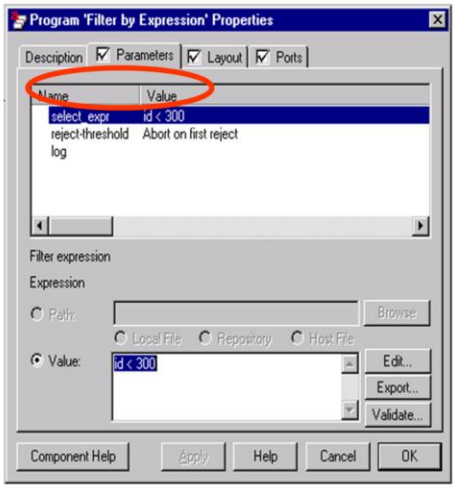
- For each record on the input port the 'select_expr' parameter is evaluated. If 'select_expr' evaluates true (non-zero), the input record is written to the 'out' port exactly as the input was read.
- If the 'select_expr' evaluates false (zero), the record is written to the 'deselect' port.
- The 'out' port must be connected downstream, those records meeting the 'select_expr' criteria
- The 'deselect' output may be optionally used

Filter Data



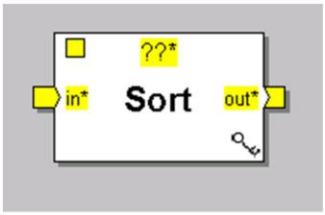
CapGemini Ab Initio Training 5

Expression Parameter

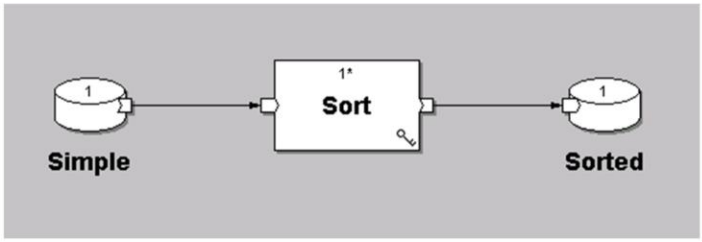


The Sort Component

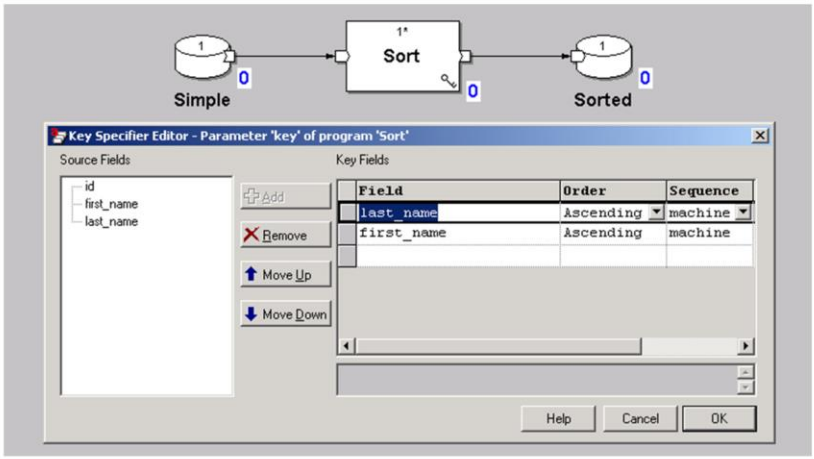
- Reads records from input port, sorts them by key, and writes the result on the output port.



Sorting



Sorting - The Key Specifier Editor



Parameters for Sort

- key

(key specifier, required)

Name(s) of the key field(s) and the sequence specifier(s) you want Sort to use when it orders data records.

- max-core

(integer, required)

Maximum memory usage in bytes. The default value of **max-core** is **100663296** (100 megabytes). When Sort reaches the number of bytes specified in the **max-core** parameter, it sorts the records it has read and writes a temporary file to disk.

Runtime Behavior of Sort

The Sort component:

- Reads the records from all the flows connected to the **in** port until it reaches the number of bytes specified in the **max-core** parameter.
- Sorts the records and writes the results to a temporary file on disk.
- Repeats this procedure until it has read all records.
- Merges all the temporary files, maintaining the sort order
- Writes the result to the **out** port Sort stores temporary files in the working directories specified by its layout.

Sort within Groups

- Sort within Groups refines the sorting of data records already sorted according to one key specifier: it sorts the records within the groups formed by the first sort according to a second key specifier.

Parameters for Sort within Groups

- major-key

(key specifier, required) :Name(s) of the key field(s) and the sequence specifier(s) by which Sort within Groups assumes input is ordered.

- minor-key

(key specifier, required) :Name(s) of the key field(s) and the sequence specifier(s) you want Sort within Groups to use when it orders data records.

- max-core

(integer, required) :Maximum memory usage in bytes before Sort within Groups stops the execution of the graph. The default value of **max-core** is **10485760** (10 megabytes).

Runtime Behavior of Sort within Groups

Sort within Groups assumes input records are sorted according to the **major-key** parameter. Sort within Groups reads data records from all the flows connected to the **in** port until it either reaches the end of a group or reaches the number of bytes specified in the **max-core** parameter. When Sort within Groups reaches the end of a group, it does the following:

- Sorts the records in the group according to the **minor-key** parameter
- Writes the results to the **out** port
- Repeats this procedure with the next group

Sort Within Groups : Example

Cust Id	Tran Ic	Tran Amt
1	668	12
1	620	23
1	648	34
1	635	43
4	823	12
4	812	34
4	870	56
4	825	78
4	882	32
7	744	43
7	788	45
7	722	56

Input data sorted by Cust Id

Minor Key

Major Key

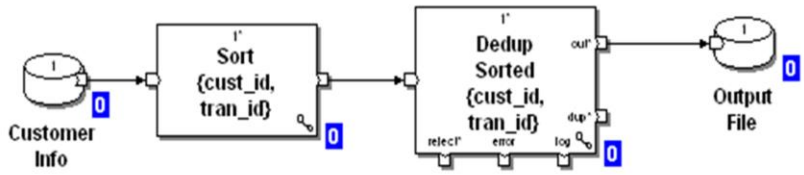
Cust Id	Tran Id	Tran Amt
1	620	23
1	635	43
1	648	34
1	668	12
4	812	34
4	823	12
4	825	78
4	870	56
4	882	32
7	722	56
7	744	43
7	788	45

Output data Sorted by Cust Id And Tran Id

CapGemini Ab Initio Training 15

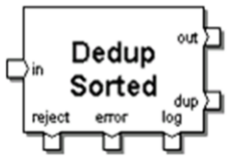
Sort Within Group

- Major Key : class
- Minor Key : roll_nbr



Dedup Sorted

- Dedup Sorted separates one specified data record in each group of data records from the rest of the records in the group.
- Dedup Sorted requires grouped input



Removing Duplicates

Input

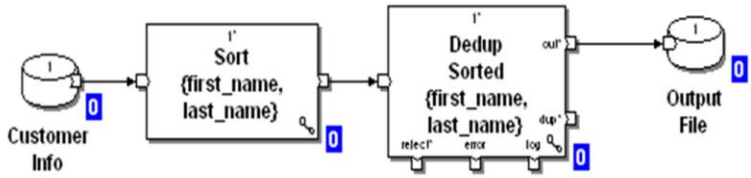
cust_id	first_name	last_name
2206	Ashish	Modi
2207	Pankaj	Parmar
2207	Pankaj	Parmar
2243	Amar	Singh
2243	Amar	Singh
2243	Amar	Singh

Occurrence of Duplicate Records in Customer Info file

Output

cust_id	first_name	last_name
2206	Ashish	Modi
2207	Pankaj	Parmar
2243	Amar	Singh

Delete Duplicates



- **Deletes duplicates from a group of records based on the key/s**
- **Data should be sorted on the same key/s before using Dedup**
- **"keep" property can be used to select either first, last or unique record from within the group**

CapGemini Ab Initio Training 19

Runtime Behavior of Dedup Sorted

- The Dedup Sorted component:
- Reads a grouped flow of records from the **in** port. If your records are not already grouped, use Sort to group them.
- Applies the expression in the **select** parameter to the records, if you have defined the **select** parameter

Runtime Behavior of Dedup Sorted

- If the expression evaluates to 0 for a particular record, Dedup Sorted does not process the record (that is, the record does not appear on any output port).
- If the expression produces NULL for a particular record, Dedup Sorted writes the record to the **reject** port and writes a descriptive error message to the error port. Dedup Sorted discards the information if you do not connect flows to the reject or error ports.
- If the expression evaluates to anything other than **0** or NULL for a particular record, Dedup Sorted processes the record. If you do not supply an expression for the **select** parameter, Dedup Sorted processes all the records on the **in** port.

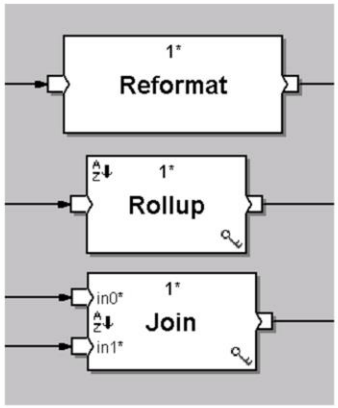
Runtime Behavior of Dedup Sorted

- Dedup Sorted considers any consecutive records with the same key value to be in the same group:
- If a group consists of one record, Dedup Sorted writes that record to the **out** port.
- If a group consists of more than one record, Dedup Sorted uses the value of the **keep** parameter to determine:
  - Which record — if any — to write to the **out** port.
  - Which record or records to write to the **dup** port.

Runtime Behavior of Dedup Sorted

- If you have chosen **unique-only** for the **keep** parameter, Dedup Sorted does not write records to the **out** port from any groups consisting of more than one record.
- Both the **out** and **dup** ports are optional; if you do not connect flows to them, Dedup Sorted discards the records.

More Complex Components

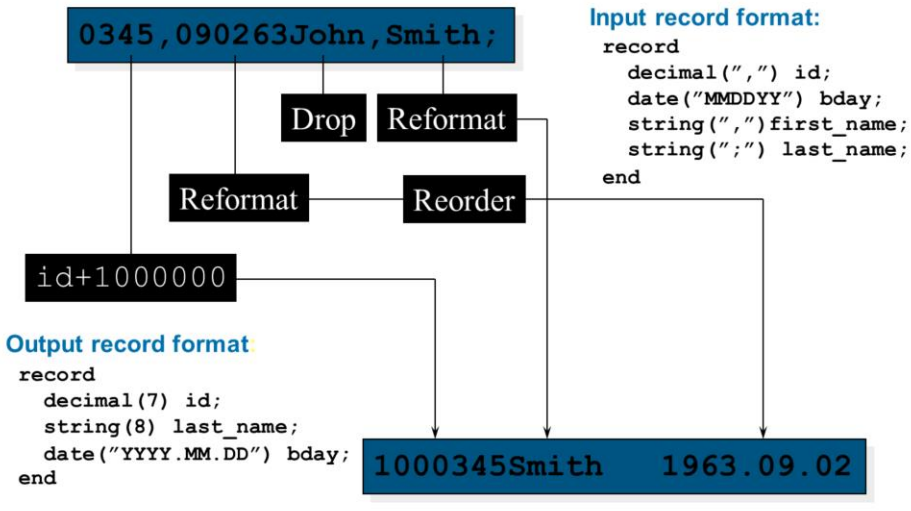


- In these components the record format metadata typically changes (goes through a transformation) from input to output

Reformat-Transform Component

- Transform components modify or manipulate data records by using one or more transform functions.
- Reformat: Changes the record format of your data by dropping fields or by using DML expressions to add fields, combine fields, or modify the data.

Data Transformation



Input record format:

```
record
  decimal(",") id;
  date("MMDDYY") bday;
  string(",")first_name;
  string(";") last_name;
end
```

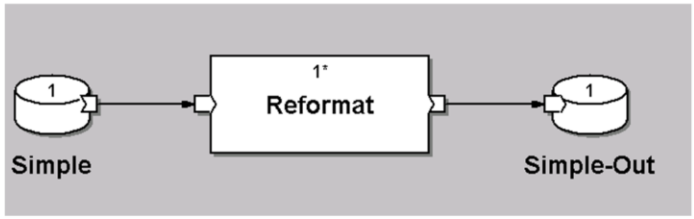
Output record format

```
record
  decimal(7) id;
  string(8) last_name;
  date("YYYY.MM.DD") bday;
end
```

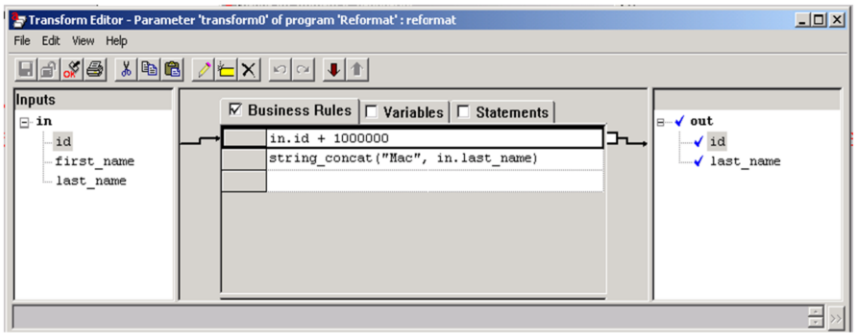
CapGemini Ab Initio Training 26

The Reformat Component

- Reads records from input port, reformats each according to a transform function (optional in the case of the Reformat Component), and writes the result records to the output (out0) port.
- Additional output ports (out1, ...) can be created by adjusting the count parameter.

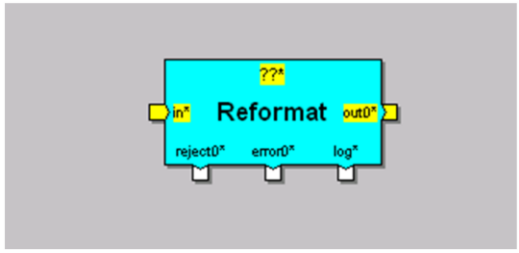


The Transform Function Editor



Reformat

- Reformat with 5 ports



About Transform Functions

- A *transform function* (or *transform*) is the logic that drives data transformation — most commonly, transform functions express record reformatting logic. In general, however, you can use transform functions in data cleansing, record merging, and record aggregation.

About Transform Functions

- To be more specific, a transform function is a collection of business rules, local variables, and statements. The transform expresses the connections between the rules, variables, and statements, as well as the connections between these elements and the input and output fields.
- Transform functions are always associated with transform components; these are components that have a **transform** parameter: the Aggregate, Denormalize Sorted, Fuse, Join, Match Sorted, MultiReformat, Normalize, Reformat, Rollup, and Scan components.

About Transform Functions

- Each component that has a **transform** parameter:
- Determines the values that are passed to the transform function
- Interprets the results of the transform function

Runtime Behavior of Reformat

- The *n* in **out***n* gives each **out** port a unique number. Each **out***n* port has a corresponding **reject***n* and **error***n* port.
- The Reformat component:
- Reads records from the **in** port.
- If you supply an expression for the **select** parameter, the expression filters the records on the **in** port:
  - If the expression evaluates to **0** for a particular record, Reformat does not process the record, which means that the record does not appear on any output port.
  - If the expression produces NULL for any record, Reformat writes a descriptive error message and stops execution of the graph.

Runtime Behavior of Reformat

- If the expression evaluates to anything other than **0** or NULL for a particular record, Reformat processes the record. If you do not supply an expression for the **select** parameter, Reformat processes all the records on the **in** port.
- Passes the records to the transform functions, calling the transform function on each port, in order, for each record, beginning with **out** port **0** and progressing through **out** port count - 1.
- Writes the results to the **out** ports.

Parameters for Reformat

Reformat :Parameters

- count
- limit
- log
- ramp
- reject-threshold
- select
- Transform n

Parameters for Reformat

- count :(integer, required)

  Integer from 1 to 20 that sets the number of each of the following. The default is **1**.
  - **out** ports
  - **reject** ports
  - **error** ports
  - **transform** parameters

Parameters for Reformat

- Transform: (filename or string, optional)

  Either the name of the file, or a transform string, containing a transform function corresponding to an **out** port; *n* represents the number of an **out** port .Transform functions for Reformat should have one input and one output.

- select :(expression, optional)

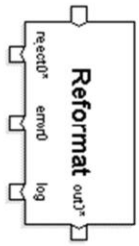
  Filter for data records before reformatting.

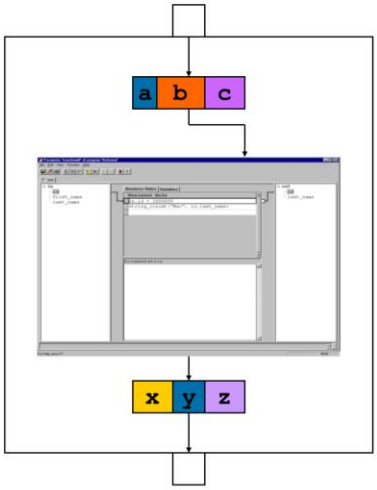
Parameters for Reformat

- limit : (integer, required)

  A number representing reject events .When the **reject-threshold** parameter is set to **Use ramp/limit**, the component uses the values of the **ramp** and **limit** parameters in a formula to determine the component's tolerance for reject events. Default is **0**.

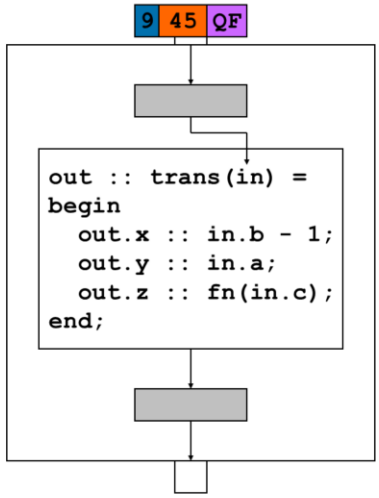
A Look Inside the Reformat Component





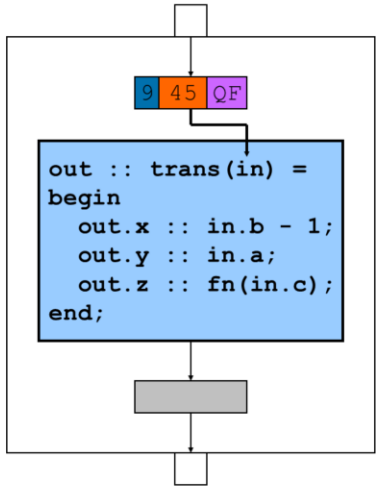
CapGemini Ab Initio Training 39

A Record arrives at the input port

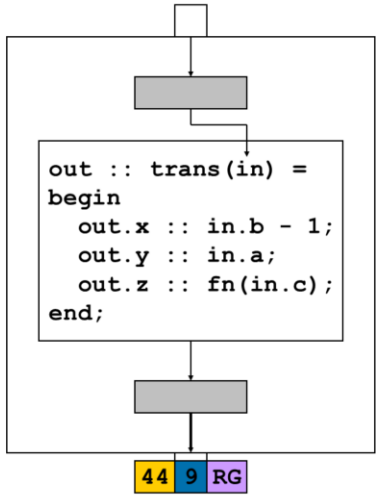


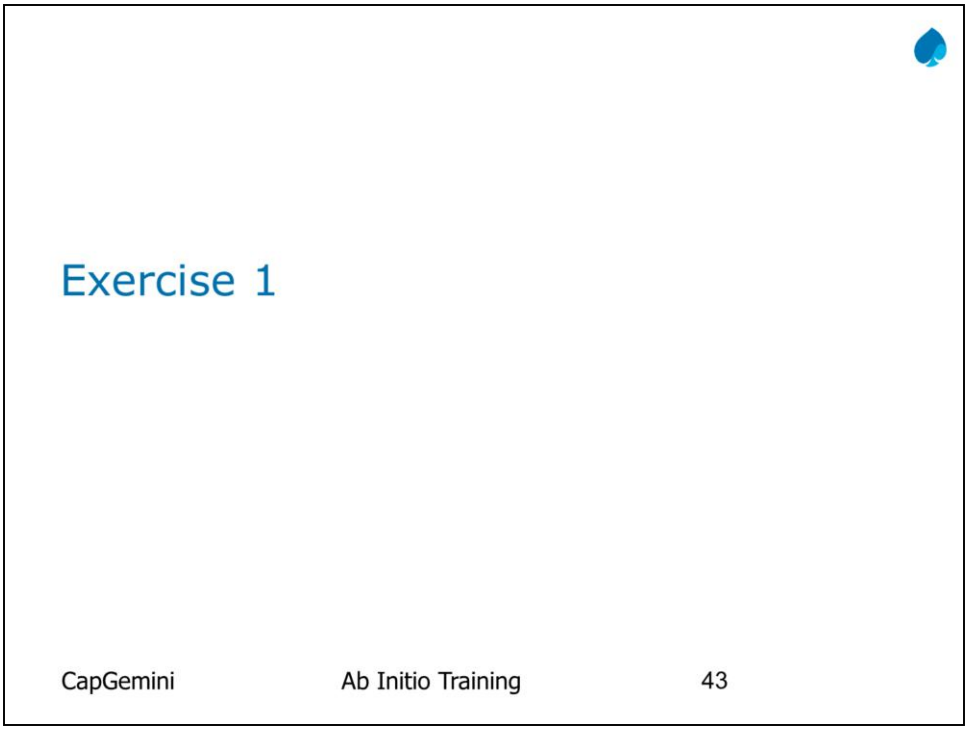
CapGemini Ab Initio Training 40

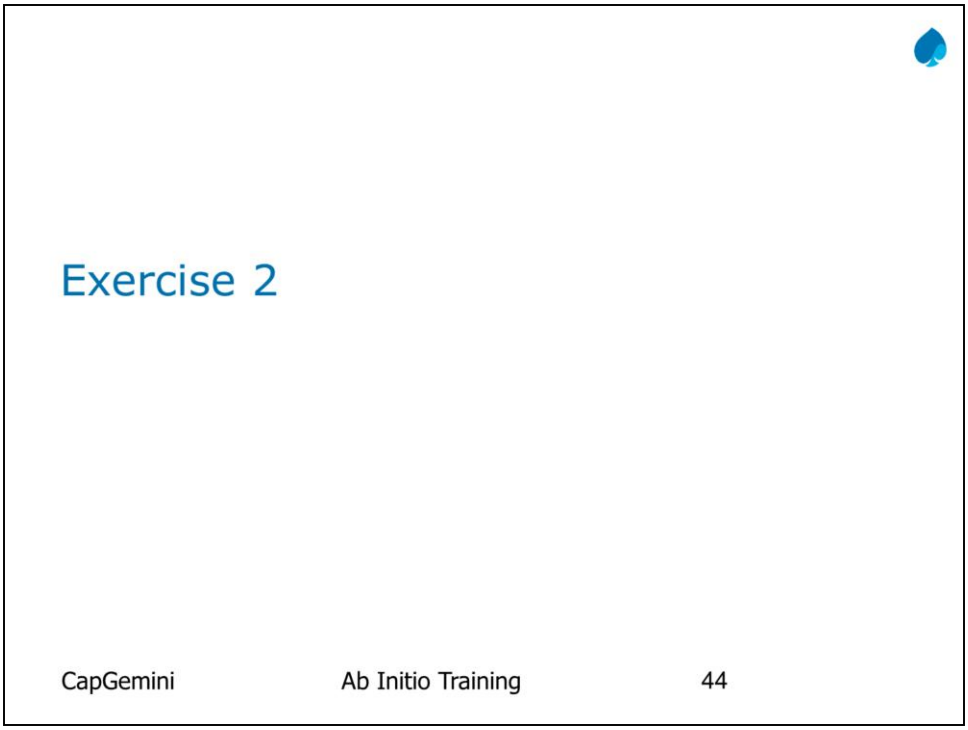
The Transformation Function is evaluated



The result record is written to the output port of the component




Exercise 1
CapGemini
Ab Initio Training
43


Exercise 2
CapGemini
Ab Initio Training
44


Thank You
End of Session 3
CapGemini
Ab Initio Training
45