Hi,

please find below some useful questions:

Informatica Scenario Based Interview Questions with Answers

1. How to generate sequence numbers using expression transformation?

Solution:
In the expression transformation, create a variable port and increment it by 1. Then assign the variable port to an output port. In the expression transformation, the ports are:
V_count=V_count+1
O_count=V_count

2. Design a mapping to load the first 3 rows from a flat file into a target?

Solution:
You have to assign row numbers to each record. Generate the row numbers either using the expression transformation as mentioned above or use sequence generatortransformation.
Then pass the output to filter transformation and specify the filter condition as O_count <=3

3. Design a mapping to load the last 3 rows from a flat file into a target?

Solution:
Consider the source has the following data.
col
a
b
c
d
e

Step1: You have to assign row numbers to each record. Generate the row numbers using the expression transformation as mentioned above and call the row number generated port as O_count. Create a DUMMY output port in the same expression transformation and assign 1 to that port. So that, the DUMMY output port always return 1 for each row.

In the expression transformation, the ports are
V_count=V_count+1
O_count=V_count
O_dummy=1

The output of expression transformation will be
col, o_count, o_dummy
a, 1, 1
b, 2, 1
c, 3, 1
d, 4, 1
e, 5, 1

Step2: Pass the output of expression transformation to aggregator and do not specify any group by condition. Create an output port O_total_records in the aggregator and assign O_count port to it. The aggregator will return the last row by default. The output of aggregator contains the DUMMY port which has value 1 and O_total_records port which has the value of total number of records in the source.

In the aggregator transformation, the ports are
O_dummy
O_count
O_total_records=O_count

The output of aggregator transformation will be
O_total_records, O_dummy
5, 1

Step3: Pass the output of expression transformation, aggregator transformation to joinertransformation and join on the DUMMY port. In the joiner transformation check the property sorted input, then only you can connect both expression and aggregator to joinertransformation.

In the joiner transformation, the join condition will be
O_dummy (port from aggregator transformation) = O_dummy (port from

expressiontransformation)

The output of joiner transformation will be
col, o_count, o_total_records
a, 1, 5
b, 2, 5
c, 3, 5
d, 4, 5
e, 5, 5

Step4: Now pass the ouput of joiner transformation to filter transformation and specify the filter condition as O_total_records (port from aggregator)-O_count(port from expression) <=2

In the filter transformation, the filter condition will be
O_total_records - O_count <=2

The output of filter transformation will be
col o_count, o_total_records
c, 3, 5
d, 4, 5
e, 5, 5

4. Design a mapping to load the first record from a flat file into one table A, the last record from a flat file into table B and the remaining records into table C?

Solution:
This is similar to the above problem; the first 3 steps are same. In the last step instead of using the filter transformation, you have to use router transformation. In the routertransformation create two output groups.

In the first group, the condition should be O_count=1 and connect the corresponding output group to table A. In the second group, the condition should be O_count=O_total_records and connect the corresponding output group to table B. The output of default group should be connected to table C.

5. Consider the following products data which contain duplicate rows.
A
B
C
C
B
D
B

Q1. Design a mapping to load all unique products in one table and the duplicate rows in another table.
The first table should contain the following output
A
D

The second target should contain the following output
B
B
B
C
C

Solution:
Use sorter transformation and sort the products data. Pass the output to an expressiontransformation and create a dummy port O_dummy and assign 1 to that port. So that, the DUMMY output port always return 1 for each row.

The output of expression transformation will be
Product, O_dummy
A, 1
B, 1
B, 1
B, 1
C, 1
C, 1
D, 1

Pass the output of expression transformation to an aggregator transformation. Check the group by on product port. In the aggreagtor, create an output port O_count_of_each_product and write an expression count(product).

The output of aggregator will be
Product, O_count_of_each_product
A, 1
B, 3
C, 2
D, 1

Now pass the output of expression transformation, aggregator transformation to joinertransformation and join on the products port. In the joiner transformation check the property sorted input, then only you can connect both expression and aggregator to joinertransformation.

The output of joiner will be
product, O_dummy, O_count_of_each_product
A, 1, 1
B, 1, 3
B, 1, 3
B, 1, 3
C, 1, 2
C, 1, 2
D, 1, 1

Now pass the output of joiner to a router transformation, create one group and specify the group condition as O_dummy=O_count_of_each_product. Then connect this group to onetable. Connect the output of default group to another table.

Q2. Design a mapping to load each product once into one table and the remaining products which are duplicated into another table.
The first table should contain the following output
A
B
C
D

The second table should contain the following output
B
B
C

Solution:
Use sorter transformation and sort the products data. Pass the output to an expressiontransformation and create a variable port,V_curr_product, and assign product port to it. Then create a V_count port and in the expression editor write IIF(V_curr_product=V_prev_product, V_count+1,1). Create one more variable port V_prev_port and assign product port to it. Now create an output port O_count port and assign V_count port to it.

In the expression transformation, the ports are
Product
V_curr_product=product
V_count=IIF(V_curr_product=V_prev_product,V_count+1,1)
V_prev_product=product
O_count=V_count

The output of expression transformation will be
Product, O_count
A, 1
B, 1
B, 2
B, 3
C, 1
C, 2
D, 1

Now Pass the output of expression transformation to a router transformation, create one group and specify the condition as O_count=1. Then connect this group to one table. Connect the output of default group to another table.

Informatica Scenario Based Questions

1. Consider the following employees data as source

employee_id, salary
10, 1000
20, 2000
30, 3000
40, 5000

Q1. Design a mapping to load the cumulative sum of salaries of employees into targettable?
The target table data should look like as

employee_id, salary, cumulative_sum
10, 1000, 1000
20, 2000, 3000
30, 3000, 6000
40, 5000, 11000

Solution:

Connect the source Qualifier to expression transformation. In the expressiontransformation, create a variable port V_cum_sal and in the expression editor write V_cum_sal+salary. Create an output port O_cum_sal and assign V_cum_sal to it.

Q2. Design a mapping to get the pervious row salary for the current row. If there is no pervious row exists for the current row, then the pervious row salary should be displayed as null.
The output should look like as

employee_id, salary, pre_row_salary
10, 1000, Null
20, 2000, 1000
30, 3000, 2000
40, 5000, 3000

Solution:

Connect the source Qualifier to expression transformation. In the expressiontransformation, create a variable port V_count and increment it by one for each row entering the expression transformation. Also create V_salary variable port and assign the expression IIF(V_count=1,NULL,V_prev_salary) to it . Then create one more variable port V_prev_salary and assign Salary to it. Now create output port O_prev_salary and assign V_salary to it. Connect the expression transformation to the target ports.

In the expression transformation, the ports will be

employee_id
salary
V_count=V_count+1
V_salary=IIF(V_count=1,NULL,V_prev_salary)
V_prev_salary=salary
O_prev_salary=V_salary

Q3. Design a mapping to get the next row salary for the current row. If there is no next row for the current row, then the next row salary should be displayed as null.
The output should look like as

employee_id, salary, next_row_salary
10, 1000, 2000
20, 2000, 3000
30, 3000, 5000
40, 5000, Null

Solution:

Step1: Connect the source qualifier to two expression transformation. In each expressiontransformation, create a variable port V_count and in the expression editor write V_count+1. Now create an output port O_count in each expression transformation. In the first expression transformation, assign V_count to O_count. In the second expressiontransformation assign V_count-1 to O_count.

In the first expression transformation, the ports will be

employee_id
salary
V_count=V_count+1
O_count=V_count

In the second expression transformation, the ports will be

employee_id
salary
V_count=V_count+1
O_count=V_count-1

Step2: Connect both the expression transformations to joiner transformation and join them on the port O_count. Consider the first expression transformation as Master and second one as detail. In the joiner specify the join type as Detail Outer Join. In the joinertransformation check the property sorted input, then only you can connect both expression transformations to joiner transformation.

Step3: Pass the output of joiner transformation to a target table. From the joiner, connect the employee_id, salary which are obtained from the first expression transformation to the employee_id, salary ports in target table. Then from the joiner, connect the salary which is obtained from the second expression transformaiton to the next_row_salary port in the target table.

Q4. Design a mapping to find the sum of salaries of all employees and this sum should repeat for all the rows.
The output should look like as

employee_id, salary, salary_sum
10, 1000, 11000
20, 2000, 11000
30, 3000, 11000
40, 5000, 11000

Solution:

Step1: Connect the source qualifier to the expression transformation. In the expressiontransformation, create a dummy port and assign value 1 to it.

In the expression transformation, the ports will be

employee_id
salary
O_dummy=1

Step2: Pass the output of expression transformation to aggregator. Create a new port O_sum_salary and in the expression editor write SUM(salary). Do not specify group by on any port.

In the aggregator transformation, the ports will be

salary
O_dummy
O_sum_salary=SUM(salary)

Step3: Pass the output of expression transformation, aggregator transformation to joinertransformation and join on the DUMMY port. In the joiner transformation check the property sorted input, then only you can connect both expression and aggregator to joinertransformation.

Step4: Pass the output of joiner to the target table.

2. Consider the following employees table as source

department_no, employee_name
20, R
10, A
10, D
20, P
10, B
10, C
20, Q
20, S

Q1. Design a mapping to load a target table with the following values from the above source?

department_no, employee_list
10, A
10, A,B
10, A,B,C
10, A,B,C,D
20, A,B,C,D,P
20, A,B,C,D,P,Q
20, A,B,C,D,P,Q,R
20, A,B,C,D,P,Q,R,S

Solution:

Step1: Use a sorter transformation and sort the data using the sort key as department_no and then pass the output to the expression transformation. In the expressiontransformation, the ports will be

department_no
employee_name
V_employee_list =
IIF(ISNULL(V_employee_list),employee_name,V_employee_list||','||employee_name)
O_employee_list = V_employee_list

Step2: Now connect the expression transformation to a target table.

Q2. Design a mapping to load a target table with the following values from the above source?

department_no, employee_list
10, A
10, A,B
10, A,B,C
10, A,B,C,D
20, P
20, P,Q
20, P,Q,R
20, P,Q,R,S

Solution:

Step1: Use a sorter transformation and sort the data using the sort key as department_no and then pass the output to the expression transformation. In the expressiontransformation, the ports will be

department_no
employee_name
V_curr_deptno=department_no
V_employee_list = IIF(V_curr_deptno! =

V_prev_deptno,employee_name,V_employee_list||','||employee_name)
V_prev_deptno=department_no
O_employee_list = V_employee_list

Step2: Now connect the expression transformation to a target table.

Q3. Design a mapping to load a target table with the following values from the above source?

department_no, employee_names
10, A,B,C,D
20, P,Q,R,S

Solution:

The first step is same as the above problem. Pass the output of expression to an aggregator transformation and specify the group by as department_no. Now connect the aggregator transformation to a target table.

....

1. Consider the following product types data as the source.

Product_id, product_type
10, video
10, Audio
20, Audio
30, Audio
40, Audio
50, Audio
10, Movie
20, Movie
30, Movie
40, Movie
50, Movie
60, Movie

Assume that there are only 3 product types are available in the source. The sourcecontains 12 records and you dont know how many products are available in each product type.

Q1. Design a mapping to select 9 products in such a way that 3 products should be selected from video, 3 products should be selected from Audio and the remaining 3products should be selected from Movie.

Solution:

Step1: Use sorter transformation and sort the data using the key as product_type.

Step2: Connect the sorter transformation to an expression transformation. In the expression transformation, the ports will be

product_id
product_type
V_curr_prod_type=product_type
V_count = IIF(V_curr_prod_type = V_prev_prod_type,V_count+1,1)
V_prev_prod_type=product_type
O_count=V_count

Step3: Now connect the expression transformaion to a filter transformation and specify the filter condition as O_count<=3. Pass the output of filter to a target table.

Q2. In the above problem Q1, if the number of products in a particular product type are less than 3, then you wont get the total 9 records in the target table. For example, see the videos type in the source data. Now design a mapping in such way that even if the number of products in a particular product type are less than 3, then you have to get those lessnumber of records from another porduc types. For example: If the number of products in videos are 1, then the reamaining 2 records should come from audios or movies. So, the total number of records in the target table should always be 9.

Solution:

The first two steps are same as above.

Step3: Connect the expression transformation to a sorter transformation and sort the data using the key as O_count. The ports in soter transformation will be

product_id
product_type
O_count (sort key)

Step3: Discard O_count port and connect the sorter transformation to an expressiontransformation. The ports in expression transformation will be

product_id
product_type
V_count=V_count+1
O_prod_count=V_count

Step4: Connect the expression to a filter transformation and specify the filter condition as O_prod_count<=9. Connect the filter transformation to a target table.

2. Design a mapping to convert column data into row data without using the normalizertransformation.
The source data looks like

col1, col2, col3
a, b, c
d, e, f

The target table data should look like

Col
a
b
c
d
e
f

Solution:

Create three expression transformations with one port each. Connect col1 from SourceQualifier to port in first expression transformation. Connect col2 from Source Qualifier to port in second expression transformation. Connect col3 from source qualifier to port in third expression transformation. Create a union transformation with three input groups and each input group should have one port. Now connect the expression transformations to the input groups and connect the union transformation to the target table.

3. Design a mapping to convert row data into column data.
The source data looks like

id, value
10, a
10, b
10, c
20, d
20, e
20, f

The target table data should look like

id, col1, col2, col3
10, a, b, c
20, d, e, f

Solution:

Step1: Use sorter transformation and sort the data using id port as the key. Then connect the sorter transformation to the expression transformation.

Step2: In the expression transformation, create the ports and assign the expressions as mentioned below.

```
id
value
V_curr_id=id
V_count= IIF(v_curr_id=V_prev_id,V_count+1,1)
V_prev_id=id
O_col1= IIF(V_count=1,value,NULL)
O_col2= IIF(V_count=2,value,NULL)
O_col3= IIF(V_count=3,value,NULL)
```

Step3: Connect the expression transformation to aggregator transformation. In the aggregator transforamtion, create the ports and assign the expressions as mentioned below.

```
id (specify group by on this port)
O_col1
O_col2
O_col3
col1=MAX(O_col1)
col2=MAX(O_col2)
col3=MAX(O_col3)
```

Stpe4: Now connect the ports id, col1, col2, col3 from aggregator transformation to the target table.

....

Take a look at the following tree structure diagram. From the tree structure, you can easilyderive the parent-child relationship between the elements. For example, B is parent of D and E.

The above tree structure data is represented in a table as shown below.

c1, c2, c3, c4
A, B, D, H
A, B, D, I
A, B, E, NULL
A, C, F, NULL
A, C, G, NULL

Here in this table, column C1 is parent of column C2, column C2 is parent of column C3, column C3 is parent of column C4.

Q1. Design a mapping to load the target table with the below data. Here you need to generate sequence numbers for each element and then you have to get the parent id. As the element "A" is at root, it does not have any parent and its parent_id is NULL.

id, element, parent_id
1, A, NULL
2, B, 1
3, C, 1
4, D, 2

5, E, 2
6, F, 3
7, G, 3
8, H, 4
9, I, 4

I have provided the solution for this problem in Oracle Sql query. If you are interested you can Click Here to see the solution.

Q2. This is an extension to the problem Q1. Let say column C2 has null for all the rows, then C1 becomes the parent of C3 and c3 is parent of C4. Let say both columns c2 and c3 has null for all the rows. Then c1 becomes the parent of c4. Design a mapping to accommodate these type of null conditions.

....

Q1. The source data contains only column 'id'. It will have sequence numbers from 1 to 1000. The source data looks like as

Id
1
2
3
4
5
6
7
8
....
1000

Create a workflow to load only the Fibonacci numbers in the target table. The target tabledata should look like as

Id
1
2
3
5
8
13
.....

In Fibonacci series each subsequent number is the sum of previous two numbers. Here assume that the first two numbers of the fibonacci series are 1 and 2.

Solution:

STEP1: Drag the source to the mapping designer and then in the Source Qualifier Transformation properties, set the number of sorted ports to one. This will sort the source data in ascending order. So that we will get the numbers in sequence as 1, 2, 3, ...1000

STEP2: Connect the Source Qualifier Transformation to the Expression Transformation. In the Expression Transformation, create three variable ports and one output port. Assign the expressions to the ports as shown below.

Ports in Expression Transformation:
```
id
v_sum = v_prev_val1 + v_prev_val2
v_prev_val1 = IIF(id=1 or id=2,1, IIF(v_sum = id, v_prev_val2, v_prev_val1) )
v_prev_val2 = IIF(id=1 or id =2, 2, IIF(v_sum=id, v_sum, v_prev_val2) )
o_flag = IIF(id=1 or id=2,1, IIF( v_sum=id,1,0) )
```

STEP3: Now connect the Expression Transformation to the Filter Transformation and specify the Filter Condition as o_flag=1

STEP4: Connect the Filter Transformation to the Target Table.

Q2. The source table contains two columns "id" and "val". The source data looks like as below

idÂ Â Â Â val
1Â Â Â Â Â a,b,c
2Â Â Â Â Â pq,m,n
3Â Â Â Â Â asz,ro,liqt

Here the "val" column contains comma delimited data and has three fields in that column.
Create a workflow to split the fields in â€œvalâ€ column to separate rows. The output should look like as below.

idÂ Â Â Â val
1Â Â Â Â Â a
1Â Â Â Â Â b
1Â Â Â Â Â c
2Â Â Â Â Â pq
2Â Â Â Â Â m
2Â Â Â Â Â n
3Â Â Â Â Â asz
3Â Â Â Â Â ro
3Â Â Â Â Â liqt

Solution:

STEP1: Connect three Source Qualifier transformations to the Source Definition

STEP2: Now connect all the three Source Qualifier transformations to the Union Transformation. Then connect the Union Transformation to the Sorter Transformation. In the sorter transformation sort the data based on Id port in ascending order.

STEP3: Pass the output of Sorter Transformation to the Expression Transformation. The ports in Expression Transformation are:

```
id (input/output port)
val (input port)
v_currend_id (variable port) = id
v_count (variable port) = IIF(v_current_id!=v_previous_id,1,v_count+1)
v_previous_id (variable port) = id
o_val (output port) = DECODE(v_count, 1,
Â Â Â Â Â Â Â SUBSTR(val, 1, INSTR(val,',',1,1)-1 ),
Â Â Â Â Â Â Â 2,
Â Â Â Â Â Â Â SUBSTR(val, INSTR(val,',',1,1)+1, INSTR(val,',',1,2)-INSTR(val,',',1,1)-1),
Â Â Â Â Â Â Â 3,
Â Â Â Â Â Â Â SUBSTR(val, INSTR(val,',',1,2)+1),
Â Â Â Â Â Â Â NULL
Â Â Â Â Â Â Â )
```

STEP4: Now pass the output of Expression Transformation to the Target definition. Connect id, o_val ports of Expression Transformation to the id, val ports of TargetDefinition.

For those who are interested to solve this problem in oracle sql, Click Here. The oracle sqlquery provides a dynamic solution where the "val" column can have varying number of fields in each row.