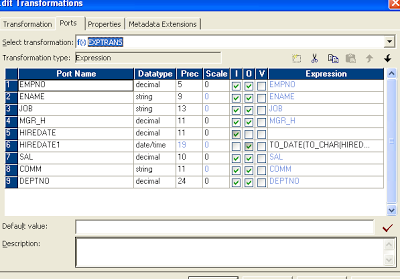
## How to****generate** **sequence** numbers using expression transformation?** Solution: In the expression transformation, create a **variable** port and increment it by 1. Then assign the variable port to an output port. In the expression transformation, the ports are: V\_count=V\_count+1 O\_count=V\_count 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 generator transformation. 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 joiner transformation 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 joiner transformation. In the joiner transformation, the join condition will be O\_dummy (port from aggregator transformation) = O\_dummy (port from expression transformation) 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 router transformation 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 expression transformation 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 thegroup 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 joiner transformation 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 joiner transformation. 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 thegroup condition as O\_dummy=O\_count\_of\_each\_product. Then connect this group to one table. 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 expression transformation 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 Convert Numeric Value to Date Format

**Senario**:Suppose you are importing a flat file emp.csv and hire\_date colummn is in numeric format, like 20101111 .Our objective is convert it to date,with a format 'YYYYMMDD'.  
  
**source**  
  EMPNO       HIRE\_DATE(numeric)            
  -------            -----------  
     1                20101111  
     2                20090909  
**target**  
EMPNO            HIRE\_DATE (date)           
 ------                   -----------  
     1                   11/11/2010  
     2                    09/09/2009

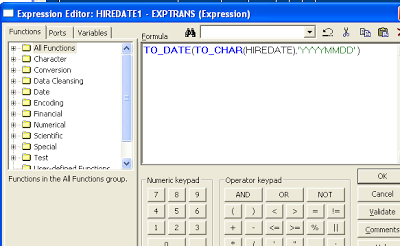
1. Connect SQF to an expression.

[](http://4.bp.blogspot.com/-6d7e1Zh_P9s/Tmiz_K0w-mI/AAAAAAAAAE0/9ogi8Rves-Y/s1600/Q28_entire_1.PNG)

 In expression make hire\_date as input only and make another port hire\_date1 as o/p port with date data type.

[](http://1.bp.blogspot.com/-PST7RSSwcWM/Tmi0dalelsI/AAAAAAAAAE4/tOoFIv33NiA/s1600/Q28_exp_2.PNG)

 In o/p port of hire\_date write condition like as below

[](http://3.bp.blogspot.com/-h84ppHR2rwM/Tmi0rST76UI/AAAAAAAAAE8/T2Hbr63aaqc/s1600/Q28_exp_condition_3.PNG)

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