## Deccan Education Society's

# Navinchandra Mehta Institute of Technology & Development

## CERTIFICATE

This is to certify that Mr. Krutarth Prasad Bodas of M.C.A. Semester I with Roll No.C23021_ has completed __All___practical's of MCAL13 **Advance Database Management System** _under my supervision in this college during the year 2023-2024.

| CO | R1: Journal | R2: Performance during lab session | R3: Implementation using different problem solving techniques | R4: Mock Viva | Attendance |
|---|---|---|---|---|---|
| CO1 | | | | | |
| CO2 | | | | | |
| CO3 | | | | | |
| CO4 | | | | | |

Practical-in-charge

Head of Department

MCA Department
(NMITD)

| MCAL13 ADBMS Lab INDEX | | | | |
|---|---|---|---|---|
| **Sr.No** | **Title** | **CO** | **Date** | **Sign** |
| 1 | Implementation of Partitions: Range, List.<br>Self-Learning Topics : Hash Partition, Composite partition | **CO1** | 05/09/2023 | |
| 2 | Analytical Queries<br>Roll_Up, CUBE, First, Last, Lead, Lag, Rank and Dense Rank<br>Self-Learning Topics: Cume_list, Percent_rank | CO3 | 06/09/2023<br>08/09/2023 | |
| 3 | Implementation of,<br>• Abstract Data Type<br>• Reference<br>Self-Learning Topics: Nested ADT, Inheritance | CO1 | 11/09/2023<br>13/09/2023 | |
| 4 | ETL Transformation with Pentaho<br>1. Copy data from Source & store to Target<br>2. Adding Sequence<br>3. Adding Calculator<br>4. Concatenation of Two Fields<br>5. Splitting of Two Fields<br>6. Number Range<br>7. String Operations<br>8. Sorting Data<br>9. Implement the Merge Join<br>10. Implement data validations on table data<br>11. Replace Strings<br>12. Splitting Fields to Rows | CO2 | 27/09/2023<br>04/10/2023<br>11/10/2023 | |
| 5 | Introduction to R,<br>Install packages<br>Loading packages<br>Data types, checking variable type,printing variable and objects (Vector, Matrix, List, Factor, Data frame, Table)<br>c-binding and rbinding<br>**Reading and Writing data:**<br>Setw(), getw(), data(),rm()<br>**Attaching and Detaching data**<br>Reading data from the console<br>Loading data from different data sources(CSV,Excel) | CO4 | 25/10/2023<br>30/10/2023 | |
| 6 | Data preprocessing techniques in R<br>Naming and Renaming variables<br>Adding a new variables | CO4 | 01/11/2023<br>01/12/2023 | |

| | | | | |
|---|---|---|---|---|
| | Dealing with missing value<br>Dealing with categorical data<br>Data reduction using subsetting | | | |
| 7 | Implementation and analysis of Linear regression through graphical methods. | CO4 | 04/12/2023 | |
| 8 | Implementation and Analysis Classification algorithms like Naïve Bayesiam, K-Nearest Neighbour, ID3, C4.5 | CO4 | 08/12/2023 | |
| 9 | Implementation and analysis of Apriori Algorithm using Market Basket Analysis | CO4 | 11/12/2023 | |
| 10 | Implementation and analysis of clustering algorithms like K-means, Agglomarative | CO4 | 12/12/2023 | |

# Practical No.: 1

# Implementation of Different Types of Partitions

## RANGE PARTITIONING

```
SQL> CREATE TABLE employee21sales_range
  2   (Salesman_id NUMBER(5),
  3     salesman_name VARCHAR2(30),
  4     sales_amount NUMBER(10),
  5     sales_date DATE)
  6     PARTITION BY RANGE(sales_date)
  7     (
  8     PARTITION sales_jan2002 VALUES LESS
THAN(TO_DATE('01/02/2002','DD/MM/YYYY')),
  9     PARTITION sales_feb2002 VALUES LESS
THAN(TO_DATE('01/03/2002','DD/MM/YYYY')),
 10     PARTITION sales_mar2002 VALUES LESS
THAN(TO_DATE('01/04/2002','DD/MM/YYYY')),
 11     PARTITION sales_apr2002 VALUES LESS
THAN(TO_DATE('01/05/2002','DD/MM/YYYY'))
 12    )
 13     ;

Table created.


SQL> SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
  2     TABLESPACE_NAME='USERS';

no rows selected


SQL> insert into employee21sales_range values(1,'Krutarth
Bodas',3000,TO_DATE('12/02/2002','DD/MM/YYYY'));

1 row created.


SQL> SELECT*FROM employee21sales_range;

SALESMAN_ID  SALESMAN_NAME   SALES_AMOUNT      SALES_DAT
--------------------- ---------------------------  -------------------------  ------------------
         1                  Krutarth Bodas            3000                 12-FEB-02


SQL> insert into employee21sales_range values(2,'Noel
Ruke',4000,TO_DATE('06/03/2002','DD/MM/YYYY'));

1 row created.
```

```
SQL> SELECT*FROM employee21sales_range;

SALESMAN_ID  SALESMAN_NAME      SALES_AMOUNT     SALES_DAT
--------------------  ----------------------------    ------------------------     ---------------
        1               Krutarth Bodas               3000             12-FEB-02
        2               Noel Ruke                    4000             06-MAR-02

SQL> insert all
  2  into employee21sales_range values(3,'Atharv
Mhabadi',5000,TO_DATE('02/04/2002','DD/MM/YYYY'))
  3  into employee21sales_range values(4,'Ganesh
Mahinnd',6000,TO_DATE('10/01/2002','DD/MM/YYYY'))
  4  into employee21sales_range values(5,'Chandresh
Chouhan',7000,TO_DATE('20/04/2002','DD/MM/YYYY'))
  5  into employee21sales_range values(6,'Aditya
Chande',8000,TO_DATE('12/02/2002','DD/MM/YYYY'))
  6  into employee21sales_range values(7,'Aman
Mishra',9000,TO_DATE('18/03/2002','DD/MM/YYYY'))
  7  into employee21sales_range values(8,'Shivtej
Patil',9500,TO_DATE('27/04/2002','DD/MM/YYYY'))
  8  select*from dual;

6 rows created.


SQL> SELECT*FROM employee21sales_range;

SALESMAN_ID   SALESMAN_NAME    SALES_AMOUNT     SALES_DAT
--------------------- ----------------------------   ------------------------     -----------------
        4               Ganesh Mahinnd             6000             10-JAN-02
        1               Krutarth Bodas             3000             12-FEB-02
        6               Aditya Chande              8000             12-FEB-02
        2               Noel Ruke                  4000             06-MAR-02
        7               Aman Mishra                9000             18-MAR-02
        3               Atharv Mhabadi             5000             02-APR-02
        5              Chandresh Chouhan           7000             20-APR-02
        8               Shivtej Patil              9500             27-APR-02

8 rows selected.

SQL> select*from employee21sales_range partition(sales_jan2002);

SALESMAN_ID   SALESMAN_NAME   SALES_AMOUNT   SALES_DAT
--------------------- -------------------------- ------------------------ --------------------
        4                 Ganesh Mahinnd            6000            10-JAN-02



SQL> select*from employee21sales_range partition(sales_feb2002);

SALESMAN_ID   SALESMAN_NAME   SALES_AMOUNT    SALES_DAT
--------------------- --------------------------  ------------------------  -------------------
        1                 Krutarth Bodas            3000             12-FEB-02
        6                 Aditya Chande             8000             12-FEB-02
```

## LIST PARTITIONING

```
SQL> CREATE TABLE KB21sales_list
  2    (salesman_id NUMBER(5),
  3     salesman_name VARCHAR2(30),
  4     sales_city VARCHAR2(30),
  5     sales_amount NUMBER(10),
  6     sales_date DATE)
  7     PARTITION BY LIST(sales_city)
  8     (
  9     PARTITION sales_west VALUES('Virar','Saphale'),
 10     PARTITION sales_Harbur VALUES('Vashi','Panvel','Juinagar'),
 11     PARTITION sales_central VALUES('Thane','Kanjurmarg'),
 12     PARTITION sales_other VALUES(DEFAULT)
 13     )
 14     enable row movement
 15     ;

Table created.



SQL> DESC KB21sales_list;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 SALESMAN_ID                                        NUMBER(5)
 SALESMAN_NAME                                      VARCHAR2(30)
 SALES_CITY                                         VARCHAR2(30)
 SALES_AMOUNT                                       NUMBER(10)
 SALES_DATE                                         DATE



SQL> SELECT TABLE_NAME,PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
  2  TABLESPACE_NAME='USERS';

no rows selected


SQL> insert into KB21sales_list values(1,'Noel
Ruke','Virar','2000',TO_DATE('12/01/2002','DD/MM/YYYY'));

1 row created.


SQL> insert into KB21sales_list values(2,'Mahesh
Kamble','Panvel','3000',TO_DATE('15/02/2002','DD/MM/YYYY'));

1 row created.
```

```
SQL> insert all
  2  into KB21sales_list values(3,'Atharv
Mhabadi','Juinagar','3500',TO_DATE('06/03/2002','DD/MM/YYYY'))
  3  into KB21sales_list values(4,'Chandresh
Chouhan','Kanjurmarg','3600',TO_DATE('08/04/2002','DD/MM/YYYY'))
  4  into KB21sales_list values(5,'Krutarth
Bodas','Thane','4000',TO_DATE('03/05/2002','DD/MM/YYYY'))
  5  into KB21sales_list values(6,'Omkar
Bodas','Saphale','4500',TO_DATE('04/06/2002','DD/MM/YYYY'))
  6  into KB21sales_list values(7,'Amogh
Bodas','Vashi','4200',TO_DATE('05/07/2002','DD/MM/YYYY'))
  7  into KB21sales_list values(8,'Aditya
Chande','Virar','3700',TO_DATE('08/09/2002','DD/MM/YYYY'))
  8  select*from dual;

6 rows created.


SQL> select * from KB21sales_list;

SALESMAN_ID  SALESMAN_NAME    SALES_CITY   SALES_AMOUNT   SALES_DAT
-----------  ---------------  -----------  -------------  ----------
     1          Noel Ruke           Virar         2000        12-JAN-02
     6          Omkar Bodas         Saphale       4500        04-JUN-02
     8          Aditya Chande       Virar         3700        08-SEP-02
     2          Mahesh Kamble       Panvel        3000        15-FEB-02
     3          Atharv Mhabadi      Juinagar      3500        06-MAR-02
     7          Amogh Bodas         Vashi         4200        05-JUL-02
     4          Chandresh Chouhan   Kanjurmarg    3600        08-APR-02
     5          Krutarth Bodas      Thane         4000        03-MAY-02


8 rows selected.

SQL> insert all
  2  into KB21sales_list values(9,'Shivtej Patil','Charni
Road','2500',TO_DATE('06/07/2002','DD/MM/YYYY'))
  3  into KB21sales_list values(10,'Pranav Jadhav','Grant
Road','2600',TO_DATE('16/08/2002','DD/MM/YYYY'))
  4  select*from dual;

2 rows created.



SQL> SELECT*FROM KB21sales_list partition(sales_west);

SALESMAN_ID  SALESMAN_NAME    SALES_CITY   SALES_AMOUNT   SALES_DAT
-----------  ---------------  -----------  -------------  ----------

     1          Noel Ruke           Virar         2000        12-JAN-02

     6          Omkar Bodas         Saphale       4500        04-JUN-02

     8          Aditya Chande       Virar         3700        08-SEP-02
```

```
SQL> SELECT*FROM KB21sales_list partition(sales_harbur);

SALESMAN_ID    SALESMAN_NAME      SALES_CITY    SALES_AMOUNT       SALES_DAT
-------------- ------------------ ------------- ------------------ -----------
     2            Mahesh Kamble       Panvel        3000            15-FEB-02

     3            Atharv Mhabadi      Juinagar      3500            06-MAR-02

     7            Amogh Bodas         Vashi         4200            05-JUL-02
```

```
SQL> SELECT*FROM KB21sales_list partition(sales_central);

SALESMAN_ID   SALESMAN_NAME      SALES_CITY    SALES_AMOUNT     SALES_DAT
------------- ------------------ ------------- ---------------- ----------
     4          Chandresh Chouhan    Kanjurmarg      3600         08-APR-02

     5          Krutarth Bodas       Thane           4000         03-MAY-02
```

```
SQL> SELECT*FROM KB21sales_list partition(sales_other);

 SALESMAN_ID   SALESMAN_NAME      SALES_CITY    SALES_AMOUNT     SALES_DAT
-------------- ------------------ ------------- ---------------- ------------
     9              Shivtej Patil     Charni Road     2500         06-JUL-02

     10             Pranav Jadhav     Grant Road      2600         16-AUG-02
```

# HASH PARTIONING

```
SQL> CREATE TABLE sales_krutarth21
   (salesman_id NUMBER(5),
   salesman_name VARCHAR2(30),
   sales_amount NUMBER(10),
   week_no NUMBER(2) )
   PARTITION BY HASH(salesman_id)
   PARTITIONS 4
   ;

Table created.



SQL> insert into sales_krutarth21 values(101,'Krutarth',45000,12);

1 row created.

SQL> insert into sales_krutarth21 values(102,'Noel',46000,13);

1 row created.

SQL> insert into sales_krutarth21 values(103,'Atharv',47000,11);

1 row created.


SQL> select*from sales_krutarth21;

SALESMAN_ID   SALESMAN_NAME   SALES_AMOUNT   WEEK_NO
------------- --------------- -------------- --------
      102            Noel            46000        13
      103            Atharv          47000        11
      101            Krutarth        45000        12



SQL> insert all
  2  into sales_krutarth21 values(104,'Shivtej',48000,14)
  3  into sales_krutarth21 values(105,'Pranav',49000,17)
  4  into sales_krutarth21 values(106,'Shriraj',46000,18)
  5  into sales_krutarth21 values(107,'Rushikesh',45000,22)
  6  into sales_krutarth21 values(108,'Uzma',47000,27)
  7  into sales_krutarth21 values(109,'Divya',44000,24)
  8  select*from dual;

6 rows created.
```

```
SQL> select*from sales_krutarth21;

SALESMAN_ID  SALESMAN_NAME  SALES_AMOUNT   WEEK_NO
-------------------- -------------------------- ------------------------- ----------------
      108            Uzma                47000             27
      104            Shivtej             48000             14
      102            Noel                46000             13
      103            Atharv              47000             11
      105            Pranav              49000             17
      107            Rushikesh            45000            22
      109            Divya                44000            24
      101            Krutarth             45000            12
      106            Shriraj              46000            18
9 rows selected.

SQL> SELECT TABLE_NAME, PARTITION_NAME FROM USER_TAB_PARTITIONS WHERE
  TABLESPACE_NAME='SYSTEM';


     TABLE_NAME                    PARTITION_NAME
------------------------------   ------------------------------------------------

SALES_KRUTARTH21                   SYS_P401

SALES_KRUTARTH21                   SYS_P402

SALES_KRUTARTH21                   SYS_P403

SALES_KRUTARTH21                   SYS_P404


SQL> select*from sales_krutarth21 partition(SYS_P401);

SALESMAN_ID  SALESMAN_NAME   SALES_AMOUNT    WEEK_NO
-------------------- ---------------------------- --------------------------- -----------------
      108            Uzma                   47000            27


SQL> select*from sales_krutarth21 partition(SYS_P402);

SALESMAN_ID  SALESMAN_NAME   SALES_AMOUNT    WEEK_NO
-------------------- ---------------------------- --------------------------- -----------------
      104             Shivtej             48000              14
```

```
SQL> select*from sales_krutarth21 partition(SYS_P403);

SALESMAN_ID  SALESMAN_NAME    SALES_AMOUNT     WEEK_NO
-------------------- ---------------------------- --------------------------- -----------------
      102                 Noel                  46000              13
      103                 Atharv                47000              11
      105                 Pranav                49000              17
      107                 Rushikesh              45000              22
      109                 Divya                  44000              24
```

```
SQL> select*from sales_krutarth21 partition(SYS_P404);

SALESMAN_ID  SALESMAN_NAME    SALES_AMOUNT     WEEK_NO
-------------------- ---------------------------- --------------------------- -----------------
      101               Krutarth                 45000             12
      106               Shriraj                  46000             18
```

# COMPOSITE PARTITIONING

```
SQL> CREATE TABLE purchase
  2  (
  3  purchase_no NUMBER,
  4  purchase_date DATE,
  5  Product NUMBER,
  6  Quantity NUMBER
  7  )
  8  PARTITION BY RANGE(purchase_date)
  9  SUBPARTITION BY HASH(Product)SUBPARTITIONS 4
 10  (
 11  PARTITION order1 VALUES LESS
 12  THAN(TO_DATE('01/02/2023','DD/MM/YYYY')),
 13  PARTITION order2 VALUES LESS
 14  THAN(TO_DATE('01/03/2023','DD/MM/YYYY')),
 15  PARTITION order3 VALUES LESS
 16  THAN(TO_DATE('01/04/2023','DD/MM/YYYY')),
 17  PARTITION order4 VALUES LESS
 18  THAN(TO_DATE('01/05/2023','DD/MM/YYYY'))
 19  );

Table created.


SQL> SELECT TABLE_NAME,PARTITION_NAME FROM USER_TAB_PARTITIONS
  2  WHERE TABLESPACE_NAME='SYSTEM';
```

```
TABLE_NAME                    PARTITION_NAME
------------------------------ --------------------------------------------------
PURCHASE                      ORDER1

PURCHASE                      ORDER2

PURCHASE                      ORDER3

PURCHASE                      ORDER4


SQL> INSERT INTO purchase
  2  VALUES(11,TO_DATE('11/02/2023','DD/MM/YYYY'),101,5);

1 row created.


SQL> INSERT INTO purchase
  2  VALUES(12,TO_DATE('31/01/2023','DD/MM/YYYY'),102,4);

1 row created.


SQL> INSERT INTO purchase
  2  VALUES(13,TO_DATE('03/03/2023','DD/MM/YYYY'),103,2);

1 row created.
```

```
SQL> SELECT * FROM purchase PARTITION(order1);

 PURCHASE_NO   PURCHASE_    PRODUCT    QUANTITY
---------------------- -------------------- ----------------- -------------------
        12               31-JAN-23        102             4


SQL> SELECT * FROM purchase PARTITION(order2);

PURCHASE_NO   PURCHASE_    PRODUCT    QUANTITY
---------------------- -------------------- ----------------- -------------------
        11               11-FEB-23        101            5


SQL> SELECT * FROM purchase PARTITION(order3);

PURCHASE_NO   PURCHASE_    PRODUCT    QUANTITY
---------------------- -------------------- ----------------- -------------------
        13              03-MAR-23        103             2


SQL> SELECT * FROM purchase PARTITION(order4);

no rows selected
```

# Practical No.: 2

## Implementation of Analytical Queries

```
SQL> CREATE TABLE Employee21data
   (Emp_no NUMBER(5),
   Dep_no NUMBER(5),
   Birth_date DATE,
   Salary NUMBER(10),
   Comm NUMBER(8),
   Job VARCHAR2(30)
   )
   ;

Table created.

SQL> insert all
   into Employee21data values(101,10,TO_DATE('03/09/2002','DD/MM/YYYY'),30000,2000,'Manager')
   into Employee21data values(102,11,TO_DATE('13/07/2003','DD/MM/YYYY'),23000,1000,'Designer')
   into Employee21data values(103,11,TO_DATE('22/08/2003','DD/MM/YYYY'),23500,1000,'Designer')
   into Employee21data values(104,12,TO_DATE('21/04/2004','DD/MM/YYYY'),22000,800,'Tester')
   into Employee21data values(105,12,TO_DATE('17/11/2003','DD/MM/YYYY'),21000,800,'Tester')
   into Employee21data values(106,13,TO_DATE('04/02/2004','DD/MM/YYYY'),18000,400,'Developer')
   into Employee21data values(107,13,TO_DATE('04/12/2002','DD/MM/YYYY'),19000,800,'Developer')
   into Employee21data values(108,14,TO_DATE('15/06/2005','DD/MM/YYYY'),12000,500,'Sales')
  into Employee21data values(109,14,TO_DATE('19/02/2004','DD/MM/YYYY'),12500,550,'Sales')
  into Employee21data values(110,14,TO_DATE('14/11/2005','DD/MM/YYYY'),13500,600,'Sales')
  select*from dual;

10 rows created.


SQL> select*from Employee21data;

  EMP_NO     DEP_NO     BIRTH_DAT     SALARY      COMM       JOB
---------------- -------------- -------------------- ----------------  ------------   -------------
     101         10         03-SEP-02       30000       2000       Manager

     102         11         13-JUL-03       23000       1000       Designer

     103         11         22-AUG-03       23500       1000       Designer

     104         12         21-APR-04       22000       800        Tester

     105         12         17-NOV-03       21000       800        Tester

     106         13         04-FEB-04       18000       400        Developer

     107         13         04-DEC-02       19000       800        Developer

     108         14         15-JUN-05       12000       500        Sales

     109         14         19-FEB-04       12500       550        Sales

     110         14         14-NOV-05       13500       600        Sales

10 rows selected.
```

## ROLLUP

```
SQL> SELECT Dep_no,Job,count(*),sum(salary)
  2  from Employee21data
  3  group by rollup(Dep_no,Job);

   DEP_NO       JOB                 COUNT(*)   SUM(SALARY)
---------------- ----------------------- ---------------------- -------------------------
       10          Manager               1          30000
       11          Designer              2          46500
       12          Tester                2          43000
       13          Developer             2          37000
       14          Sales                 3          38000
       10                                1          30000
       11                                2          46500
       12                                2          43000
       13                                2          37000
       14                                3          38000
                                        10          194500
11 rows selected.
```

```
SQL> select Dep_no,Job,sum(salary)
  2  from Employee21data
  3  where Dep_no in(10,11)
  4  group by Dep_no, rollup(Job);

  DEP_NO              JOB                SUM(SALARY)
----------------  ------------------------------------  -----------------------
       10                 Manager              30000
       11                 Designer             46500
       10                                      30000
       11                                      46500
```

```
SQL> select Dep_no,Job,sum(salary)
  2  from Employee21data
  3  where Dep_no in(12,13,14)
  4  group by Dep_no, rollup(Job);

DEP_NO                  JOB              SUM(SALARY)
----------------  ------------------------------------  -----------------------
       12                 Tester               43000
       13                 Developer            37000
       14                 Sales                38000
       12                                      43000
       13                                      37000
       14                                      38000
6 rows selected.
```

```
SQL> SELECT Dep_no,Job,count(*),sum(salary)
  2  from Employee21data
  3  group by job,rollup(Dep_no);

   DEP_NO       JOB                COUNT(*)      SUM(SALARY)
---------------- ----------------------- ---------------------- -------------------------
      10        Manager                1            30000
      11        Designer               2            46500
      12        Tester                 2            43000
      13        Developer              2            37000
      14        Sales                  3            38000
                Manager                1            30000
                Designer               2            46500
                Tester                 2            43000
                Developer              2            37000
                Sales                  3            38000
10 rows selected.
```

## CUBE

```
SQL> SELECT Dep_no,Job,count(*),sum(salary)
  2  from Employee21data
  3  group by cube(Dep_no,Job);

  DEP_NO          JOB             COUNT(*)      SUM(SALARY)
---------------- ----------------------- ---------------------- -------------------------
                                    10           194500
                   Sales            3            38000
                   Tester           2            43000
                   Manager          1            30000
                   Designer         2            46500
                   Developer        2            37000
     10                             1            30000
     10            Manager          1            30000
     11                             2            46500
     11            Designer         2            46500
     12                             2            43000

   DEP_NO          JOB             COUNT(*)      SUM(SALARY)
---------------- ----------------------- ---------------------- -------------------------
     12             Tester              2           43000
     13                                 2           37000
     13             Developer           2           37000
     14                                 3           38000
     14              Sales              3           38000

16 rows selected.
```

# RANK

```
SQL> select Emp_no,Dep_no,salary,comm,
  2  rank() over(partition by Dep_no order by salary)as Rank
  3  from Employee21data;

    EMP_NO     DEP_NO     SALARY       COMM       RANK
---------- ---------- ---------- ---------- ----------
       101         10      30000       2000          1
       102         11      23000       1000          1
       103         11      23500       1000          2
       105         12      21000        800          1
       104         12      22000        800          2
       106         13      18000        400          1
       107         13      19000        800          2
       108         14      12000        500          1
       109         14      12500        550          2
       110         14      13500        600          3

10 rows selected.
```

# DENSE RANK

```
SQL> select Emp_no,Dep_no,salary,comm,
  2  dense_rank() over(partition by Dep_no order by salary)as Rank
  3  from Employee21data;

EMP_NO     DEP_NO     SALARY       COMM       RANK
---------- ---------- ---------- ---------- ----------
       101         10      30000       2000          1
       102         11      23000       1000          1
       103         11      23500       1000          2
       105         12      21000        800          1
       104         12      22000        800          2
       106         13      18000        400          1
       107         13      19000        800          2
       108         14      12000        500          1
       109         14      12500        550          2
       110         14      13500        600          3

10 rows selected.
```

## LEAD

```
SQL> select Emp_no,Birth_date,
  2  lead(Birth_date,1) over(order by Birth_date) as "next"
  3  from Employee21data;

    EMP_NO     BIRTH_DAT          next
---------------- ---------------------- -----------------
       101          03-SEP-02        04-DEC-02
       107          04-DEC-02        13-JUL-03
       102          13-JUL-03        22-AUG-03
       103          22-AUG-03        17-NOV-03
       105          17-NOV-03        04-FEB-04
       106          04-FEB-04        19-FEB-04
       109          19-FEB-04        21-APR-04
       104          21-APR-04        15-JUN-05
       108          15-JUN-05        14-NOV-05
       110          14-NOV-05

10 rows selected.
```

## LAG

```
SQL> select Emp_no,Birth_date,
  2  lag(Birth_date,1) over(order by Birth_date) as "Previous"
  3  from Employee21data;

    EMP_NO      BIRTH_DAT           Previous
----------------- ----------------------  ----------------------
       101           03-SEP-02
       107           04-DEC-02          03-SEP-02
       102           13-JUL-03          04-DEC-02
       103           22-AUG-03          13-JUL-03
       105           17-NOV-03          22-AUG-03
       106           04-FEB-04          17-NOV-03
       109           19-FEB-04          04-FEB-04
       104           21-APR-04          19-FEB-04
       108           15-JUN-05          21-APR-04
       110           14-NOV-05          15-JUN-05

10 rows selected.
```

## FIRST

```
SQL> select Dep_no,salary,
  2  max(salary)keep(DENSE_RANK FIRST ORDER BY salary desc)
  3  over(PARTITION BY Dep_no)"max"
  4  from Employee21data;

    DEP_NO     SALARY        max
---------------- ---------------- -----------------
        10         30000        30000
        11         23000        23500
        11         23500        23500
        12         22000        22000
        12         21000        22000
        13         18000        19000
        13         19000        19000
        14         12000        13500
        14         12500        13500
        14         13500        13500

10 rows selected.
```

## LAST

```
SQL> select Dep_no,salary,
  2  min(salary)keep(DENSE_RANK LAST ORDER BY salary desc)
  3  over(PARTITION BY Dep_no)"min"
  4  from Employee21data;

    DEP_NO      SALARY          min
----------------- ------------------    -------------
        10           30000         30000
        11           23000         23000
        11           23500         23000
        12           22000         21000
        12           21000         21000
        13           18000         18000
        13           19000         18000
        14           12000         12000
        14           12500         12000
        14           13500         12000

10 rows selected.
```

## Practical No.: 3

## Implementation of ORDBMS Concepts like ADT,Reference

```
SQL> create type KB_name As object
  2  (
  3  fname varchar(20),
  4  mname varchar(20),
  5  Iname varchar(20)
  6  );
  7  /

Type created.


SQL> create type KBB_address As object
  2  (
  3  street varchar(20),
  4  city varchar(20),
  5  pincode number(10)
  6  );
  7  /

Type created.


SQL> create table Friend1
  2  (
  3  c_id number(5) primary key,
  4  c_name KB_name,
  5  c_add KBB_address,
  6  c_phno number(10)
  7  );

Table created.


SQL> insert into Friend1
  2  values(1,KB_name('Krutarth','P','Bodas'),
  3  KBB_address('Charai','Thane',400602),7249183848);

1 row created.


SQL> insert into Friend1
  2  values(2,KB_name('Noel','D','Ruke'),
  3  KBB_address('Gimavhne','Dapoli',415712),7499752165);

1 row created.
```

```
SQL> select*from Friend1;

     C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
        1
KB_NAME('Krutarth', 'P', 'Bodas')
KBB_ADDRESS('Charai', 'Thane', 400602)
7249183848


     C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
        2
KB_NAME('Noel', 'D', 'Ruke')
KBB_ADDRESS('Gimavhne', 'Dapoli', 415712)
7499752165



SQL> insert into Friend1
  2  values(3,KB_name('Atharv','S','Mhabadi'),
  3  KBB_address('Peth','Pune',415412),7499752888);

1 row created.


SQL> insert into Friend1
  2  values(4,KB_name('Vinay','V','Wagh'),
  3  KBB_address('Taddev','Mumbai',400012),8779621917);

1 row created.


SQL> insert into Friend1
  2  values(5,KB_name('Avaneesh','S','Bagaitkar'),
  3  KBB_address('Gimavhne','Dapoli',400603),7249156165);

1 row created.
```

```
SQL> DESC Friend1;
 Name                          Null?          Type
 ----------------------------- ----------------------- ----------------------
 C_ID                          NOT NULL       NUMBER(5)
 C_NAME                                       KB_NAME
 C_ADD                                        KBB_ADDRESS
 C_PHNO                                       NUMBER(10)

SQL> set describe depth 2;

SQL> desc Friend1;
 Name                          Null?          Type
 ----------------------------- ----------------------- ----------------------
 C_ID                          NOT NULL       NUMBER(5)
 C_NAME                                       KB_NAME
   FNAME                                      VARCHAR2(20)
   MNAME                                      VARCHAR2(20)
   INAME                                      VARCHAR2(20)
 C_ADD                                        KBB_ADDRESS
   STREET                                     VARCHAR2(20)
   CITY                                       VARCHAR2(20)
   PINCODE                                    NUMBER(10)
 C_PHNO                                       NUMBER(10)



SQL> select c.c_add.street from Friend1 c where c_id=1;

C_ADD.STREET
----------------------
       Charai




SQL> select*from Friend1;

     C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
-----------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
-----------------------------------------------------------------------------
  C_PHNO
----------------------
        1
KB_NAME('Krutarth', 'P', 'Bodas')
KBB_ADDRESS('Charai', 'Thane', 400602)
7249183848
```

```
C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
         2
KB_NAME('Noel', 'D', 'Ruke')
KBB_ADDRESS('Gimavhne', 'Dapoli', 415712)
7499752165


      C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
         3
KB_NAME('Atharv', 'S', 'Mhabadi')
KBB_ADDRESS('Peth', 'Pune', 415412)
7499752888


      C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
         4
KB_NAME('Vinay', 'V', 'Wagh')
KBB_ADDRESS('Taddev', 'Mumbai', 400012)
8779621917


      C_ID
 ---------------------
  C_NAME(FNAME, MNAME, INAME)
--------------------------------------------------------------------------------
  C_ADD(STREET, CITY, PINCODE)
--------------------------------------------------------------------------------
  C_PHNO
----------------------
         5
KB_NAME('Avaneesh', 'S', 'Bagaitkar')
KBB_ADDRESS('Gimavhne', 'Dapoli', 400603)
7249156165
```

```
SQL> select c_name from Friend1;

C_NAME(FNAME, MNAME, INAME)
-----------------------------------------------------
KB_NAME('Krutarth', 'P', 'Bodas')
KB_NAME('Noel', 'D', 'Ruke')
KB_NAME('Atharv', 'S', 'Mhabadi')
KB_NAME('Vinay', 'V', 'Wagh')
KB_NAME('Avaneesh', 'S', 'Bagaitkar')



SQL> select c_id,c.c_name.Iname from Friend1 c;

  C_ID      C_NAME.INAME
----------  -------------------------
     1             Bodas
     2             Ruke
     3            Mhabadi
     4            Wagh
     5            Bagaitkar


SQL> select c_id,c.c_name.fname from Friend1 c;

  C_ID     C_NAME.FNAME
---------- --------------------------
    1            Krutarth
    2            Noel
    3            Atharv
    4            Vinay
    5            Avaneesh



SQL> select c.c_name.fname||' '||c.c_name.mname||' '||c.c_name.Iname from Friend1 c;

C.C_NAME.FNAME||''||C.C_NAME.MNAME||''||C.C_NAME.INAME
-------------------------------------------------------------------------------------
Krutarth P Bodas
Noel D Ruke
Atharv S Mhabadi
Vinay V Wagh
Avaneesh S Bagaitkar


SQL> alter type KBB_address
  2  add attribute(name KB_name)cascade;

Type altered.
```

```
SQL> create table Friend2
  2  (c_id number(5),
  3  add1 KBB_address);

Table created.


SQL> insert into Friend2
  2  values(6,
  3  KBB_address('Charai','Thane',400601,
  4  KB_name('Kishore','D','Pawar')));

1 row created.


SQL> desc Friend2;
 Name                              Null?         Type
 ------------------------          --------  -------------------------
 C_ID                                            NUMBER(5)
 ADD1                                            KBB_ADDRESS


SQL> select c_id from Friend2;

      C_ID
    ----------
         6


SQL> select c_id,c.add1.street,c.add1.name.fname from Friend2 c;

   C_ID    ADD1.STREET        ADD1.NAME.FNAME
 ---------- ----------------------    -------------------------------
      6          Charai                     Kishore



SQL> create or replace type stud_type as object
  2  (roll_no number(5),
  3  name varchar2(30)
  4  );
  5  /

Type created.

SQL> create table Students of stud_type;

Table created.

SQL> insert into Students values(stud_type(1,'KPB'));

1 row created.
```

```
SQL> insert into Students values(stud_type(2,'NDR'));

1 row created.


SQL> select*from Students;

   ROLL_NO        NAME
 ---------------  -------------------
        1             KPB
        2             NDR


SQL> insert into Students values(stud_type(3,'ASM'));

1 row created.


SQL> insert into Students values(stud_type(4,'ASB'));

1 row created.


SQL> insert into Students values(stud_type(5,'GMK'));

1 row created.



SQL> select*from Students;

  ROLL_NO        NAME
 ---------------   -------------------
        1              KPB
        2              NDR
        3             ASM
        4             ASB
        5            GMK




SQL> select roll_no from Students;

  ROLL_NO
 ----------------
        1
        2
        3
        4
        5
```

```
SQL> select s.roll_no from Students s;

   ROLL_NO
-------------------
         1
         2
         3
         4
         5



SQL> select*from Students s
  2  where s.name='ASM';

   ROLL_NO       NAME
  ----------------  ---------------
         3            ASM



SQL> select*from Students s
  2  where s.name='GMK';

    ROLL_NO         NAME
  ----------------     ---------------
          5              GMK
```

## REF &DREF

```
SQL> CREATE OR REPLACE type Dog as object
  2    (Breed varchar2(25),
  3     Name varchar2(25),
  4     BirthDate DATE);
  5     /
Type created.
```

```
SQL> create table pet_dog of Dog;
Table created.
```

```
SQL> insert into pet_dog values (
  2  Dog('Den','Stella','02-Dec-16'));
1 row created.
```

```
SQL> insert into pet_dog values (
  2  Dog('Lab','Shiro','23-Oct-23'));
1 row created.
```

```
SQL> insert into pet_dog values (
  2  Dog('Pug','Coco','18-Apr-17'));
1 row created.
```

```
SQL> select REF(A) from pet_dog A;

REF(A)

--------------------------------------------------------------------------------

000028020917E719F0A30949C3A208A8CCA4E970A92A2420AB333248EBA81B37FC2C2E44EC
0042A0

C10000

000028020952BDAD8042B94B2FA48C084BB2D5C72F2A2420AB333248EBA81B37FC2C2E44E
C0042A0

C10001


0000280209F5C554FBA60F4284A481EF6EF08D4BC12A2420AB333248EBA81B37FC2C2E44EC
0042A0

C10002


SQL> create table Owner

  2     (OwnerName varchar2(15),

  3  PETKEPT REF Dog);

Table created.



SQL> describe Owner

 Name                                      Null?    Type

 ----------------------------------------- -------- ----------------------------

 OWNERNAME                                          VARCHAR2(15)

 PETKEPT                                            REF OF DOG



SQL> insert into Owner

  2  select'Den',

  3  REF(A)

  4  from pet_dog A

  5  where Name ='Stella';

1 row created.
```

```
SQL> insert into Owner
  2  select'Lab',
  3  REF(A)
  4  from pet_dog A
  5  where Name ='Shiro';
1 row created.

SQL> insert into Owner
  2  select'Pug',
  3  REF(A)
  4  from pet_dog A
  5  where Name ='Coco';
1 row created.

SQL> select*from Owner;
OWNERNAME
----------------------
PETKEPT
----------------------------------------------------------------------------
Den
000022020817E719F0A30949C3A208A8CCA4E970A92A2420AB333248EBA81B37FC2C2E44EC

Lab
000022020852BDAD8042B94B2FA48C084BB2D5C72F2A2420AB333248EBA81B37FC2C2E44E
C

Pug
0000220208F5C554FBA60F4284A481EF6EF08D4BC12A2420AB333248EBA81B37FC2C2E44EC
```

```
SQL> select OwnerName, DEREF(O.PETKEPT)
  2  from Owner O;

OWNERNAME
----------------------
DEREF(O.PETKEPT)(BREED, NAME, BIRTHDATE)
----------------------------------------------------------------------------
Den
DOG('Den', 'Stella', '02-DEC-16')


Lab
DOG('Lab', 'Shiro', '23-OCT-23')


Pug
DOG('Pug', 'Coco', '18-APR-17')
```

# Practical No.: 4

# Implementation of ETL transformation with Pentaho

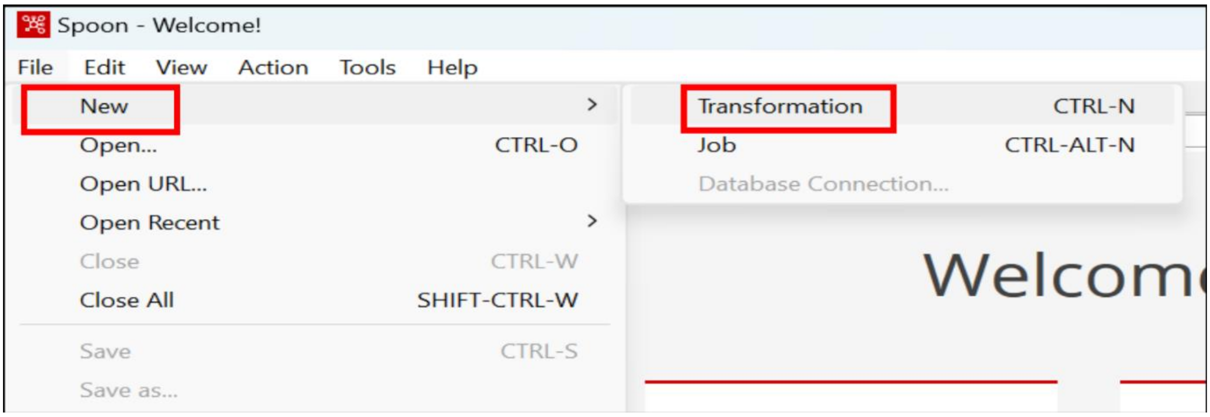
## Create Table

```
SQL> create table ADB
  2   (
  3    roll_no numeric(6),
  4    fname varchar2(10),
  5    lname varchar2(10),
  6    fees numeric(6),
  7    other_fees numeric(6),
  8    marks numeric(7)
  9    );
Table created.


SQL> select*from ADB;

  ROLL_NO    FNAME     LNAME      FEES    OTHER_FEES    MARKS
---------------- --------------- --------------- ------------  ---------------------  -------------
     11         Sam        Smith      2000        500          85
     12         Glenn      Hog        2000        800          75
     13         Ros        Taylor     2000        400          90
     14         Prithvi    Shaw       2000        650          78
     15         John       Gross      2000        950          55
     16         Pat        Rock       2000        500          85
     17         Steve      Hope       2000        800          75
     18         Tom        Lathon     2000        400          90
     19         Tonny      Stark      2000        650          78
     20          Kane      Williamson  2000       950          55

10 rows selected.
```

## Transformation 1 :

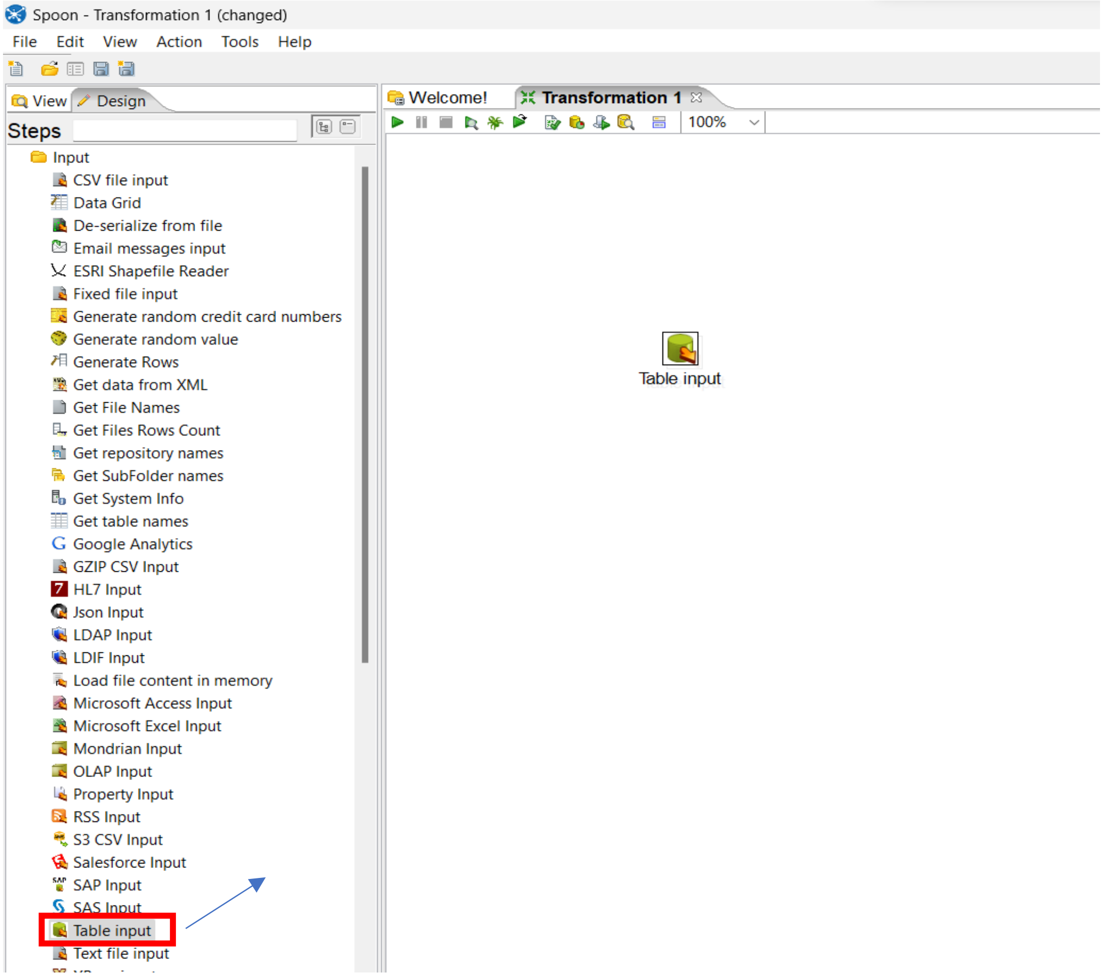
**Step 1 : -** In the data integration folder open "Spoon (Windows Batch File)".

**Step 2 : -** Go to File→New→Transformation.



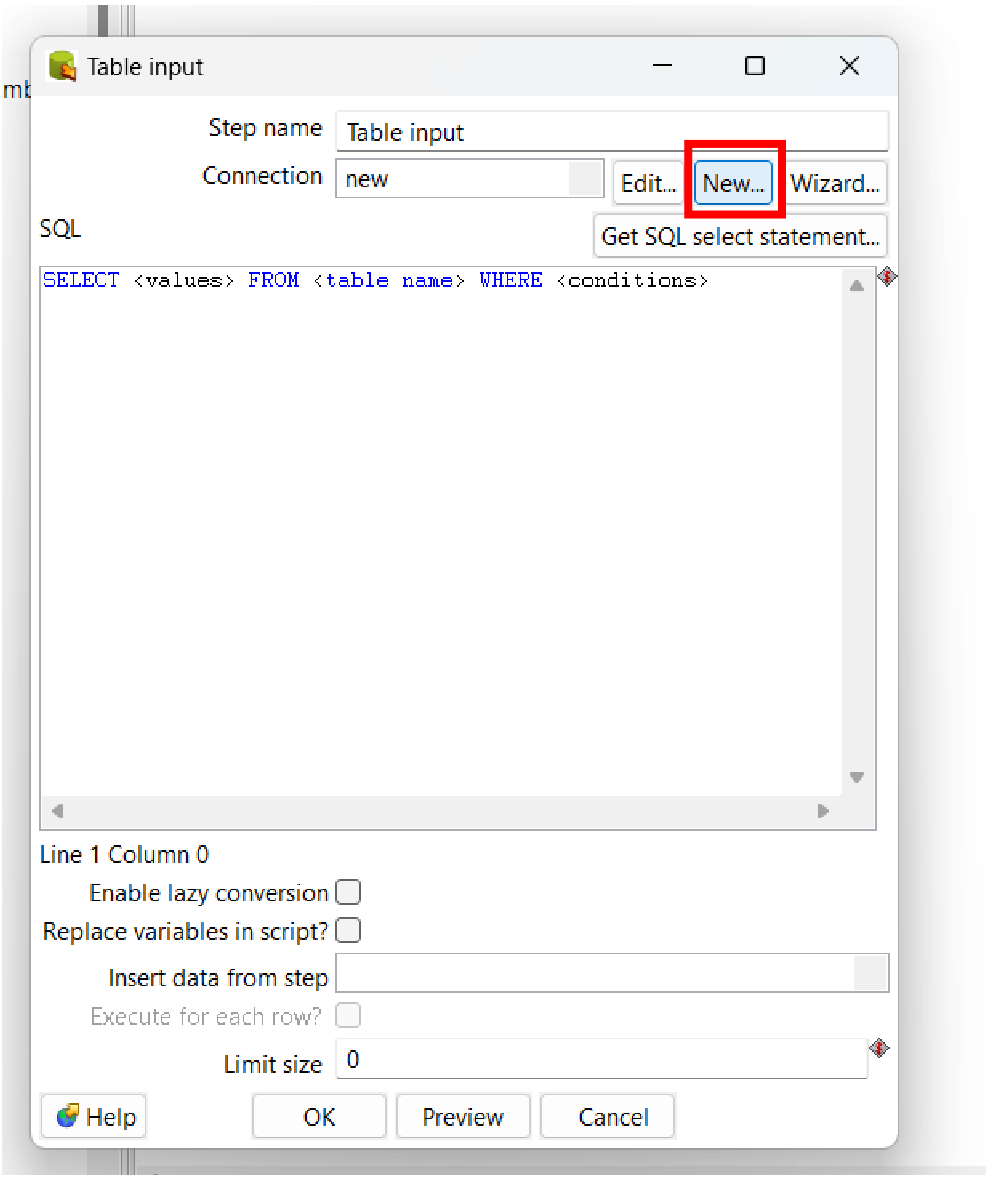
**Step 3 : -** Import SQL Table to Pentaho

Design tab →Input folder → Drag and drop Table input



Double Click On Input

Click on new



Step name: Table input

Connection: new

SQL

```
SELECT <values> FROM <table name> WHERE <conditions>
```

Line 1 Column 0

Enable lazy conversion ☐

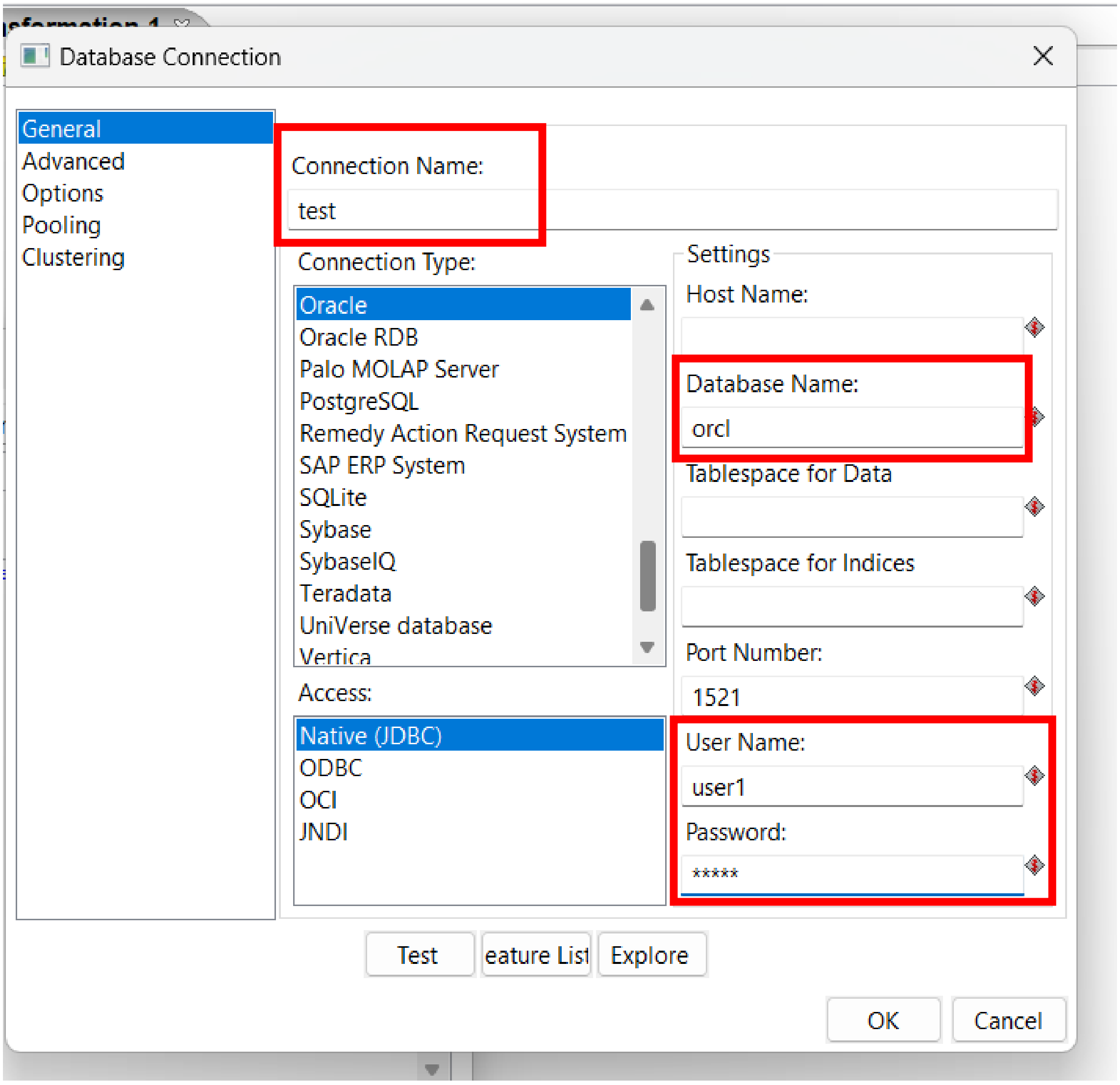
Replace variables in script? ☐

Insert data from step

Execute for each row? ☐

Limit size: 0

Connect to the Database: Fill in the details as below. Here enter User Name & Password same as your database username and password. Then click on Test.



Database Connection

Connection Name: test

Connection Type: Oracle

Access: Native (JDBC)

Settings

Host Name:

Database Name: orcl

Tablespace for Data

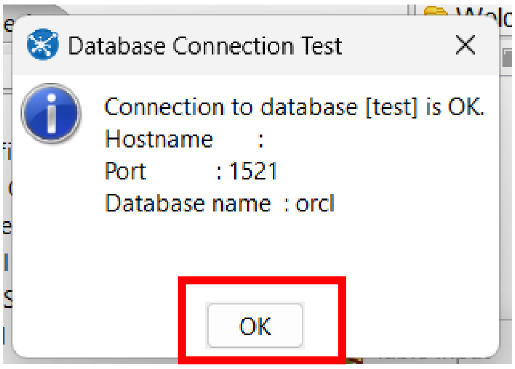
Tablespace for Indices

Port Number: 1521

User Name: user1

Password: *****

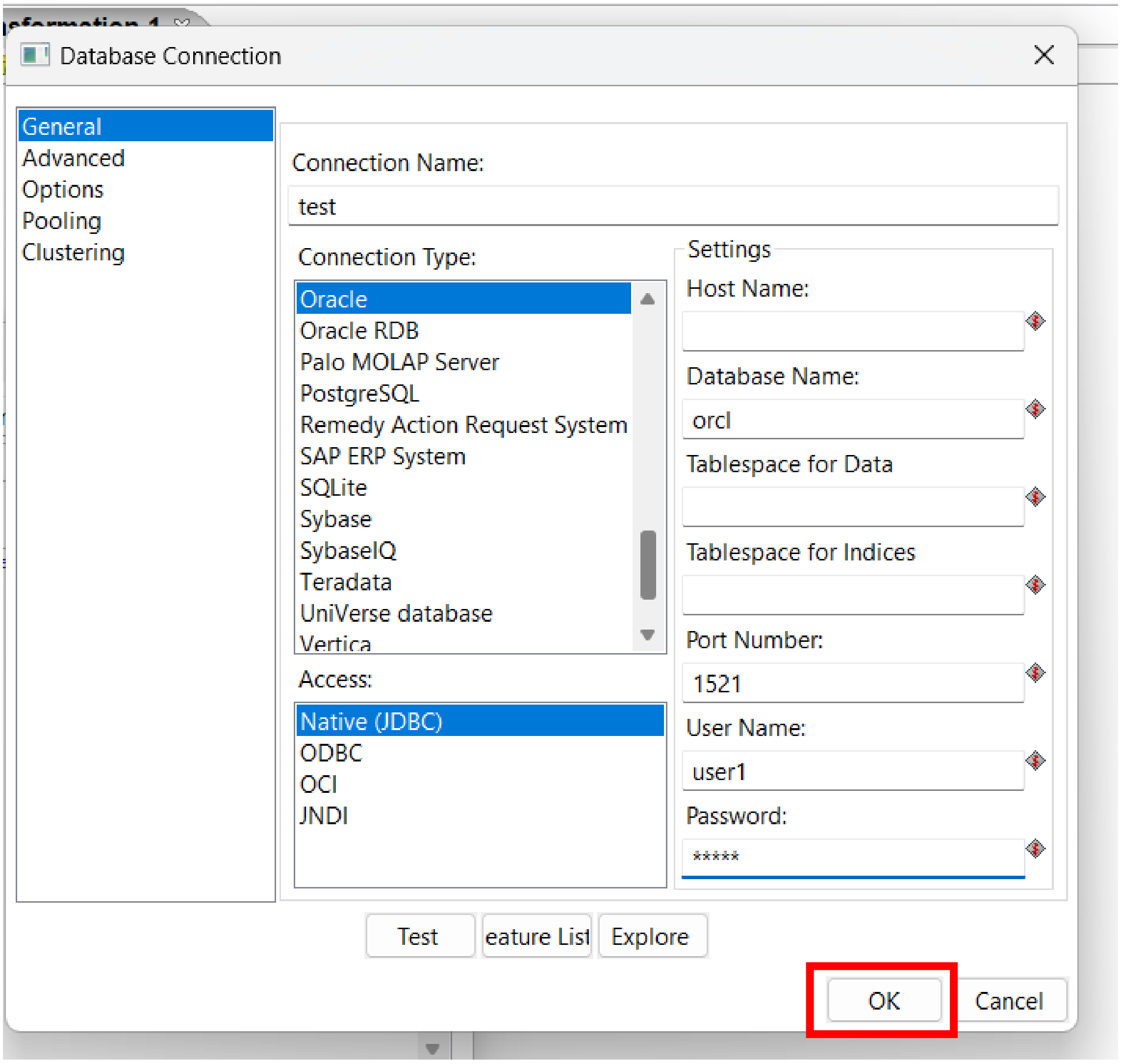
Click OK→



Database Connection Test

Connection to database [test] is OK.
Hostname :
Port : 1521
Database name : orcl

Again Ok



Database Connection

General
Advanced
Options
Pooling
Clustering

Connection Name: test

Connection Type: Oracle

Access: Native (JDBC)

Settings

Host Name:

Database Name: orcl

Tablespace for Data

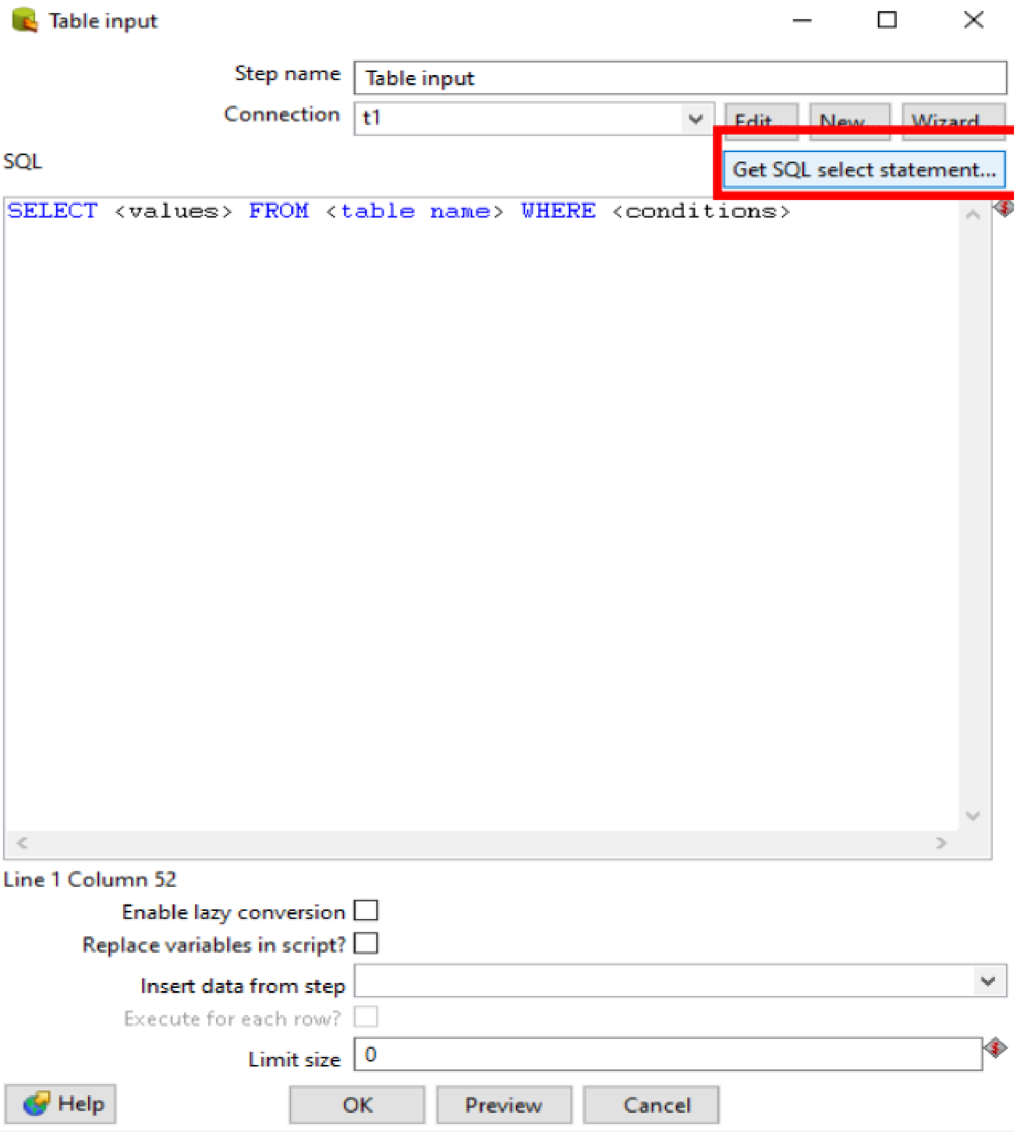
Tablespace for Indices

Port Number: 1521

User Name: user1

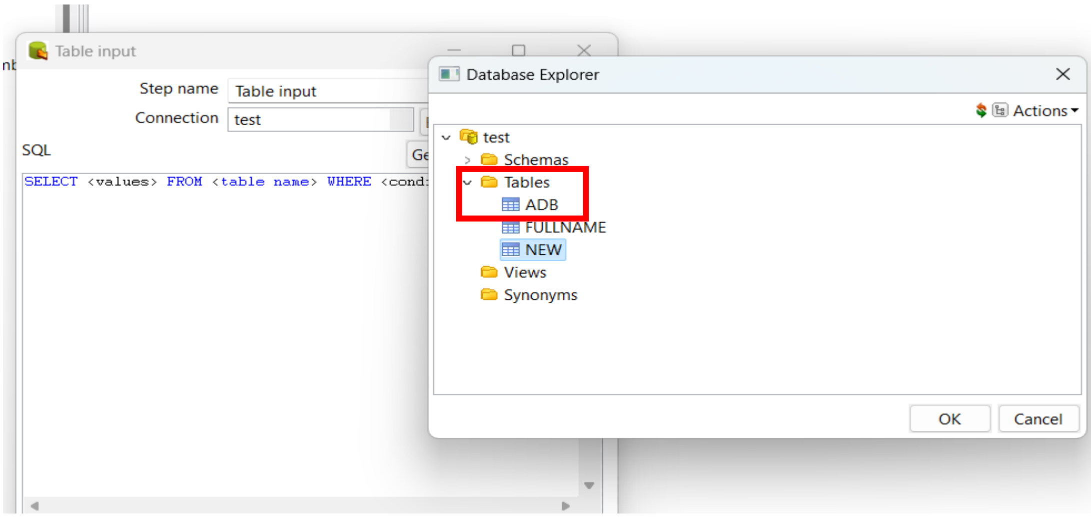
Password: *****

Get SQL select statement… in table input window

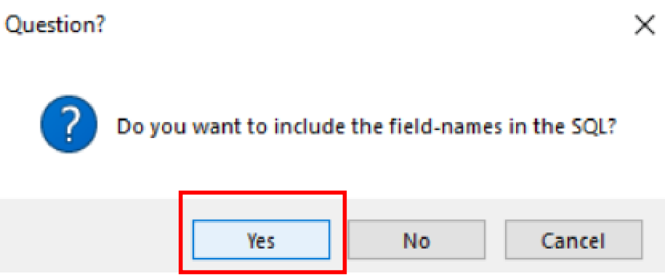


```
SELECT <values> FROM <table name> WHERE <conditions>
```

Import table: In t1, under tables, select the required table (In this case ADB).

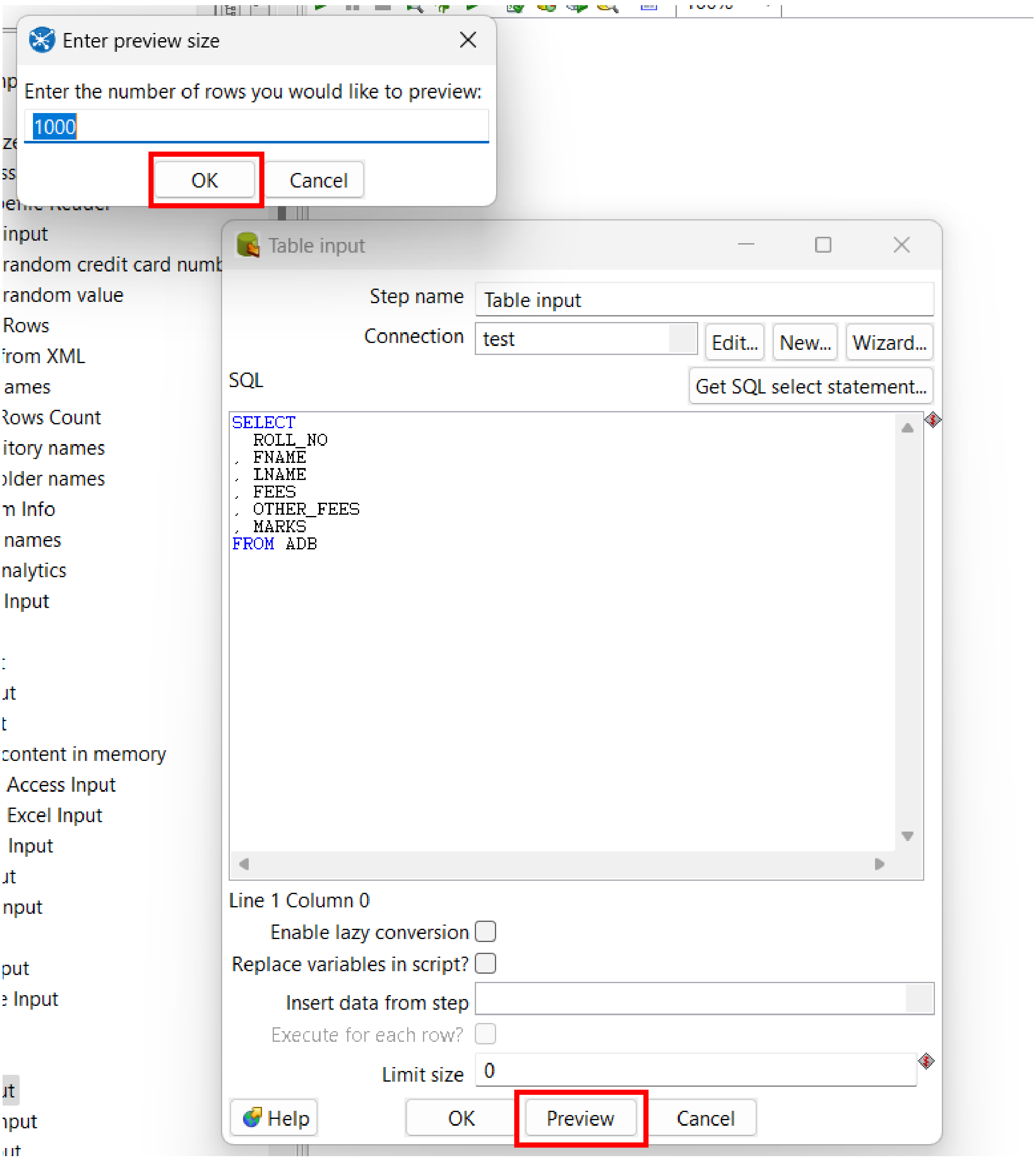


Click on OK→Click on Yes



Do you want to include the field-names in the SQL?

Preview in Table input window →OK



Enter the number of rows you would like to preview: 1000

Step name: Table input

Connection: test

SQL

```
SELECT
  ROLL_NO
, FNAME
, LNAME
, FEES
, OTHER_FEES
, MARKS
FROM ADB
```

Line 1 Column 0

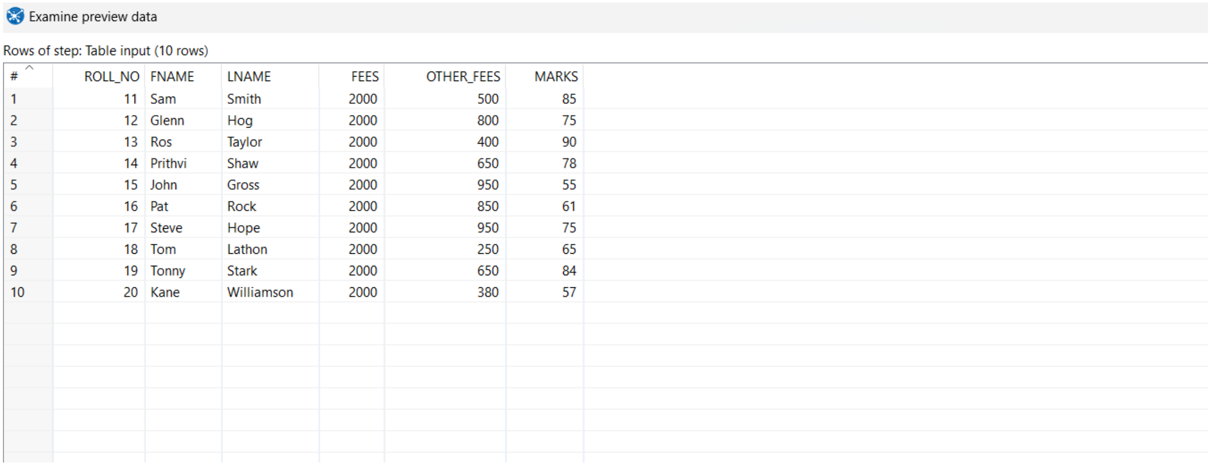
Enable lazy conversion □

Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size 0



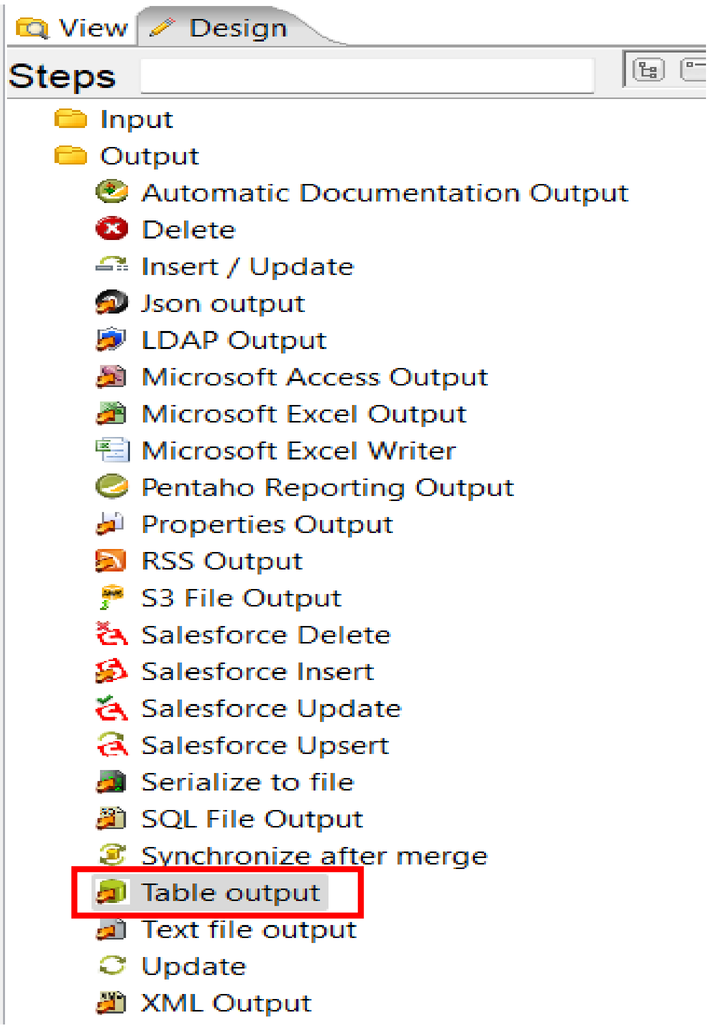
Examine preview data

Rows of step: Table input (10 rows)

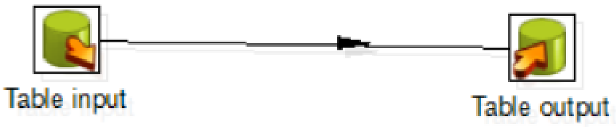
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS |
|---|---|---|---|---|---|---|
| 1 | 11 | Sam | Smith | 2000 | 500 | 85 |
| 2 | 12 | Glenn | Hog | 2000 | 800 | 75 |
| 3 | 13 | Ros | Taylor | 2000 | 400 | 90 |
| 4 | 14 | Prithvi | Shaw | 2000 | 650 | 78 |
| 5 | 15 | John | Gross | 2000 | 950 | 55 |
| 6 | 16 | Pat | Rock | 2000 | 850 | 61 |
| 7 | 17 | Steve | Hope | 2000 | 950 | 75 |
| 8 | 18 | Tom | Lathon | 2000 | 250 | 65 |
| 9 | 19 | Tonny | Stark | 2000 | 650 | 84 |
| 10 | 20 | Kane | Williamson | 2000 | 380 | 57 |

Click Close → OK

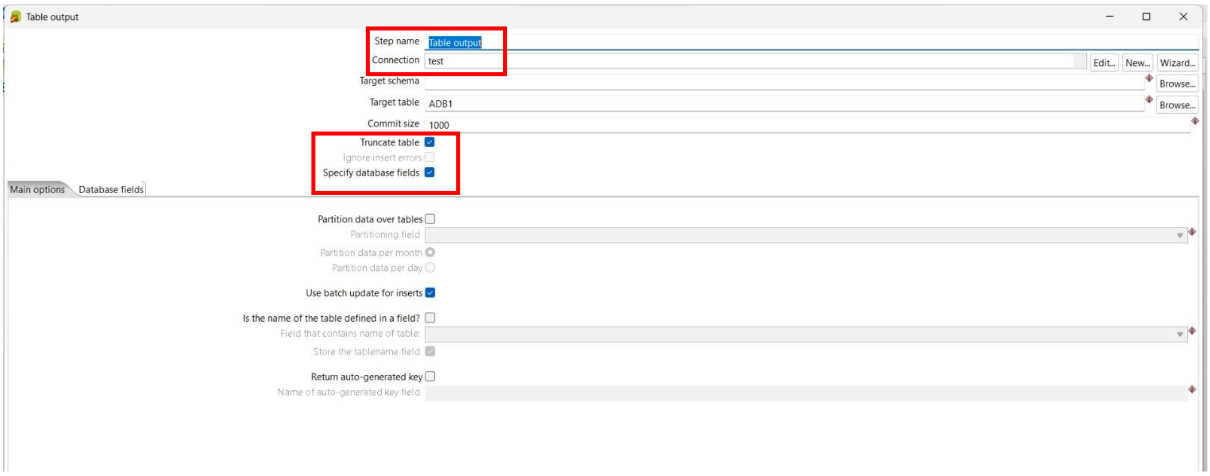
**Step 4: -** Show output: Drag and Drop Table Output



Hold the mouse pointer on Table input and select and drag the output connector to the Table output.

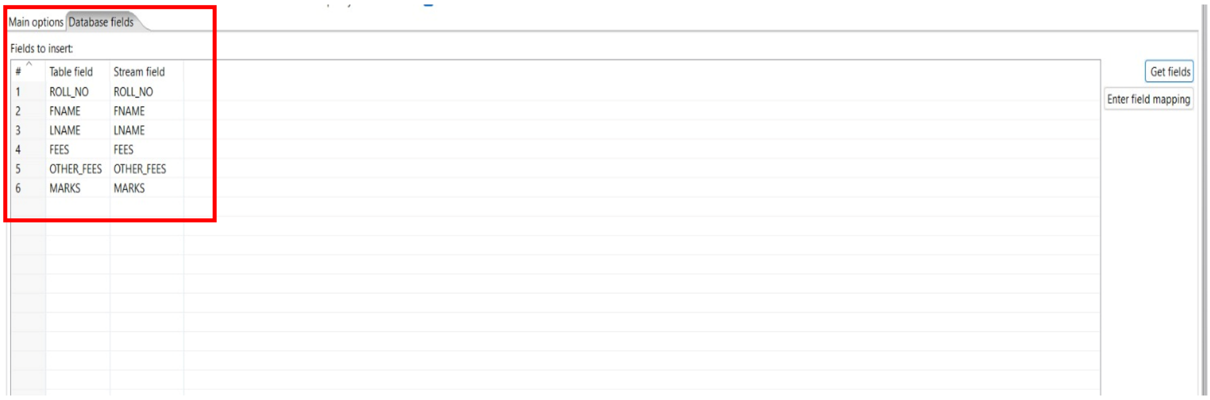


Double Click on Table Output. In the Table Output Window, give name to the Target table, check the check boxes and click on Get fields.



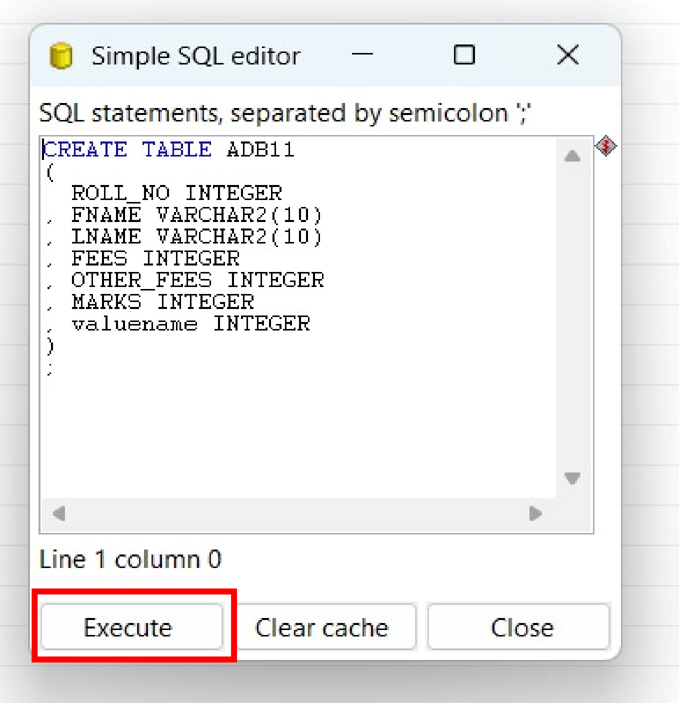
Step name: Table output
Connection: test
Target schema:
Target table: ADB1
Commit size: 1000
Truncate table ☑
Ignore insert errors □
Specify database fields ☑

Partition data over tables □
Partitioning field
Partition data per month ○
Partition data per day ○
Use batch update for inserts ☑
Is the name of the table defined in a field? □
Field that contains name of table:
Store the tablename field ☑
Return auto-generated key □
Name of auto-generated key field

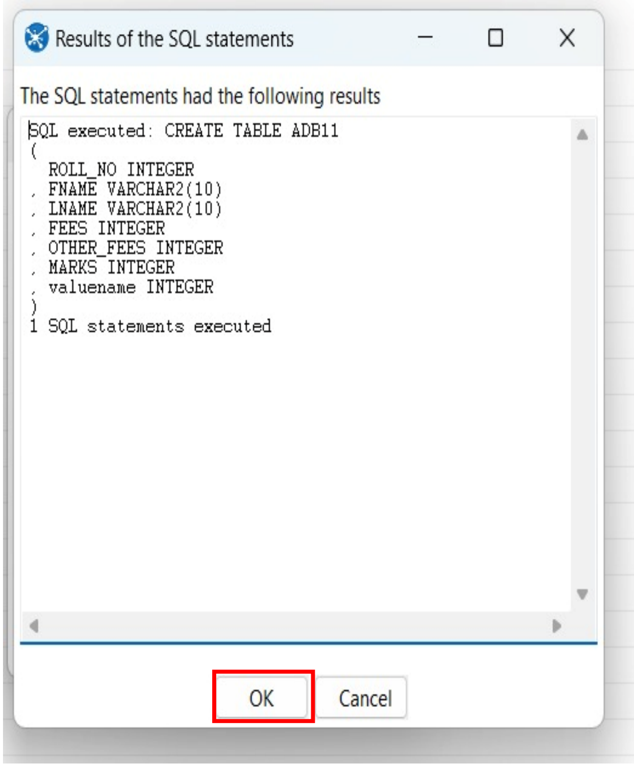


Fields to insert:

| # | Table field | Stream field |
|---|---|---|
| 1 | ROLL_NO | ROLL_NO |
| 2 | FNAME | FNAME |
| 3 | LNAME | LNAME |
| 4 | FEES | FEES |
| 5 | OTHER_FEES | OTHER_FEES |
| 6 | MARKS | MARKS |



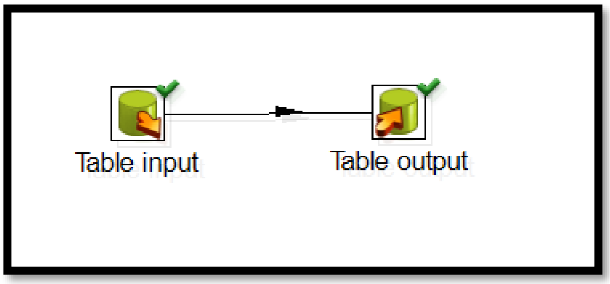
```
CREATE TABLE ADB11
(
  ROLL_NO INTEGER
, FNAME VARCHAR2(10)
, LNAME VARCHAR2(10)
, FEES INTEGER
, OTHER_FEES INTEGER
, MARKS INTEGER
, valuename INTEGER
)
;
```

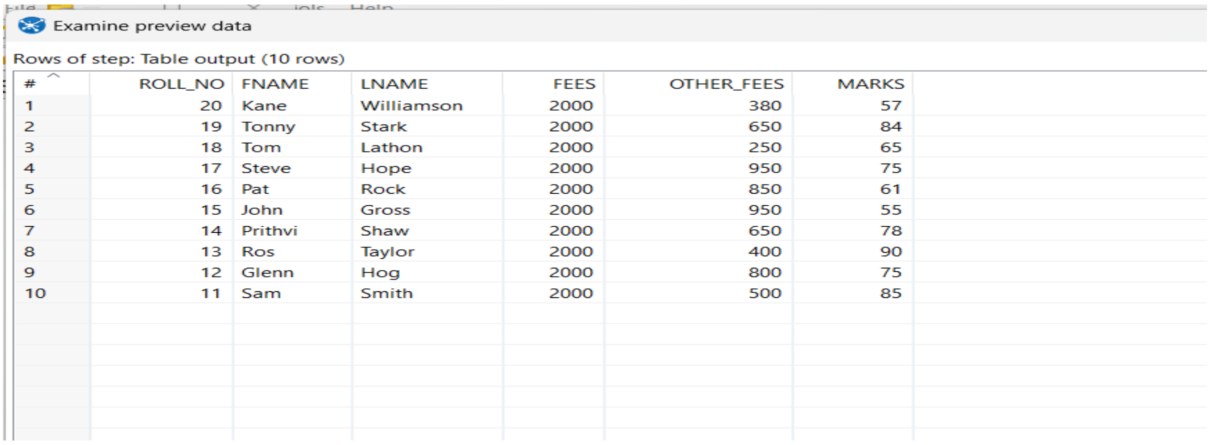


Results of the SQL statements
The SQL statements had the following results

```
SQL executed: CREATE TABLE ADB11
(
  ROLL_NO INTEGER
, FNAME VARCHAR2(10)
, LNAME VARCHAR2(10)
, FEES INTEGER
, OTHER_FEES INTEGER
, MARKS INTEGER
, valuename INTEGER
)
1 SQL statements executed
```

If the Transformation is successful, you will see green ticks.




Examine preview data

Rows of step: Table output (10 rows)

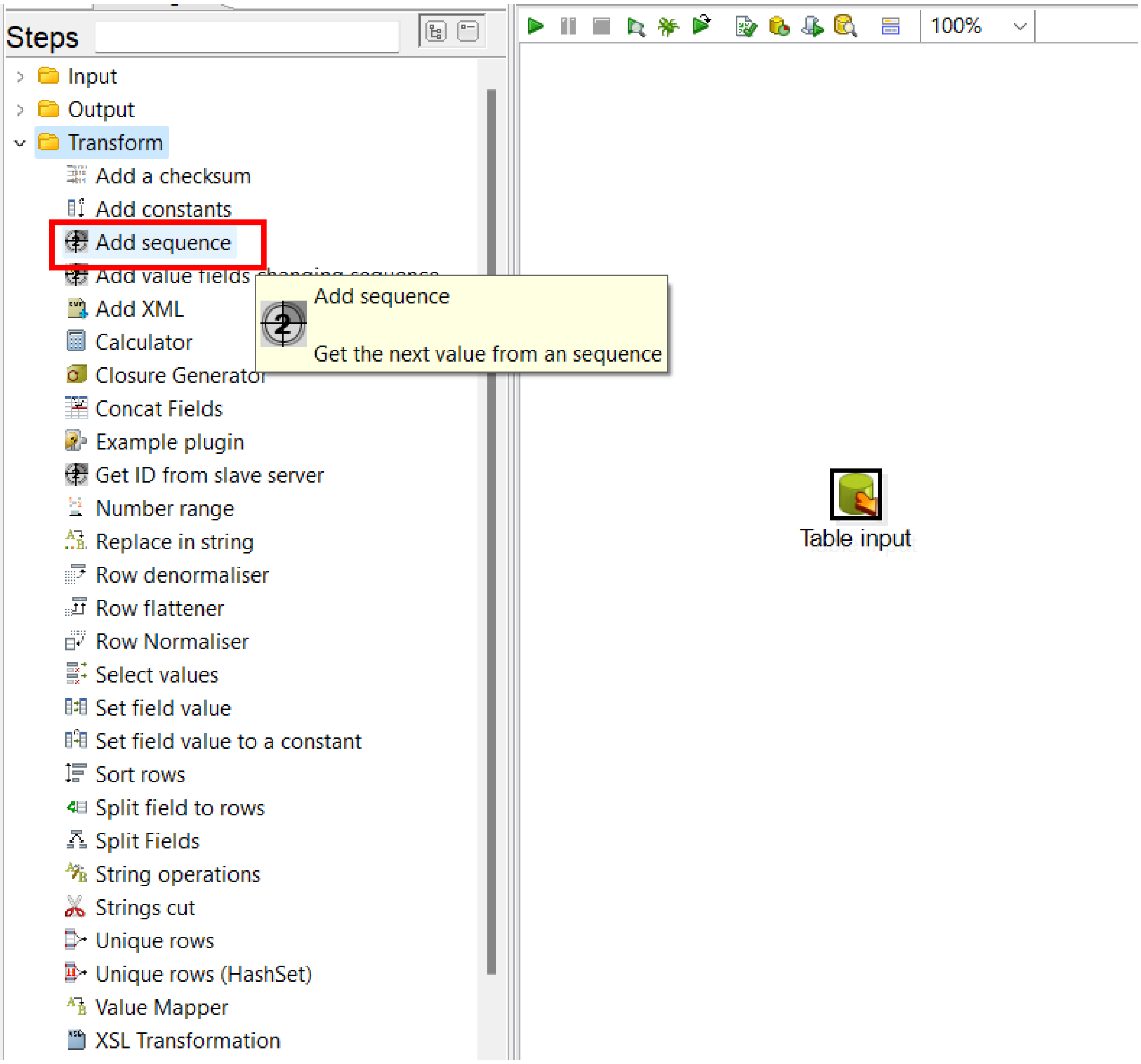
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS |
|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 |
| 6 | 15 | John | Gross | 2000 | 950 | 55 |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 |

```
SQL> select*from ADB1;

  ROLL_NO    FNAME    LNAME        FEES    OTHER_FEES    MARKS
--------------- ---------------- ------------------ ------------- ---------------------- -------------------

    11         Sam        Smith       2000         500          85

    12         Glenn      Hog         2000         800          75

    13         Ros        Taylor      2000         400          90

    14         Prithvi    Shaw        2000         650          78

    15         John       Gross       2000         950          55

    16         Pat        Rock        2000         850          61

    17         Steve      Hope        2000         950          75

    18         Tom        Lathon      2000         250          65

    19         Tonny      Stark       2000         650          84

    20         Kane       Williamson  2000         380          57


10 rows selected.
```

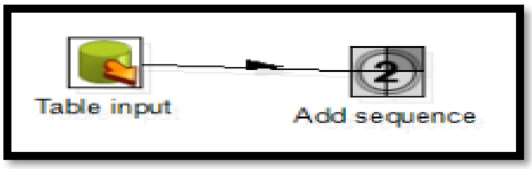
**TRANSFORMATION 2:** Add sequence.

**Step 1: -** Repeat Steps 2 and 3 from TRANSFORMATION 1.

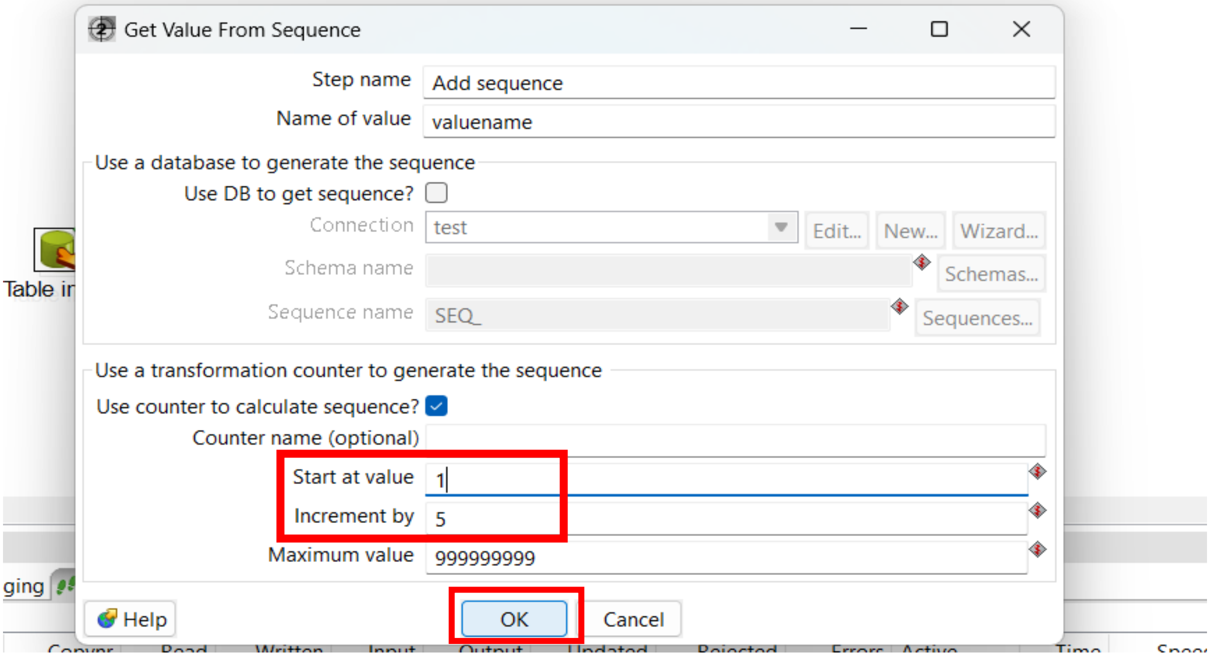
**Step 2: -** Perform transformation (Add sequence). Drag and drop Add Sequence from the transform folder under the Design tab



Hold the mouse pointer on Table input and select and drag the output connector to the Add sequence.



Double click on Add sequence and fill in the details as shown below→Click on OK.



Get Value From Sequence

Step name: Add sequence

Name of value: valuename

Use a database to generate the sequence

Use DB to get sequence? □

Connection: test

Schema name:

Sequence name: SEQ_

Use a transformation counter to generate the sequence

Use counter to calculate sequence? ☑

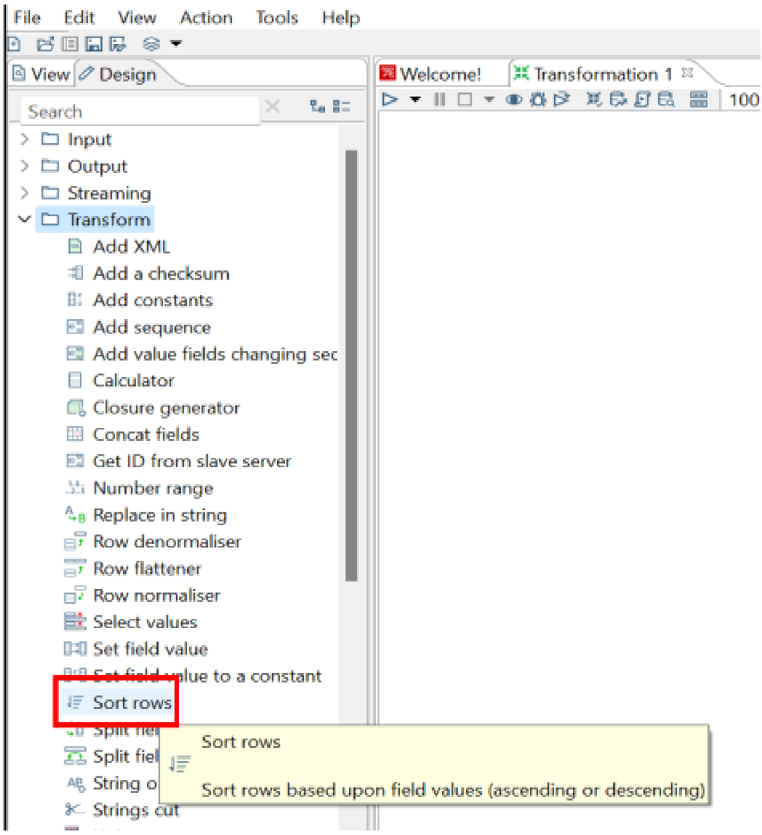
Counter name (optional):

Start at value: 1

Increment by: 5

Maximum value: 999999999

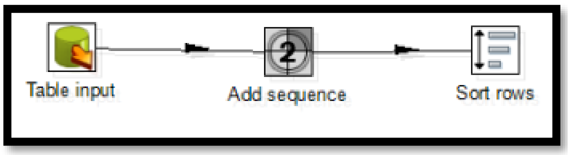
**Step 3: -** Perform transformation (Sort rows) Drag and drop Sort rows from the transform folder under the Design tab.

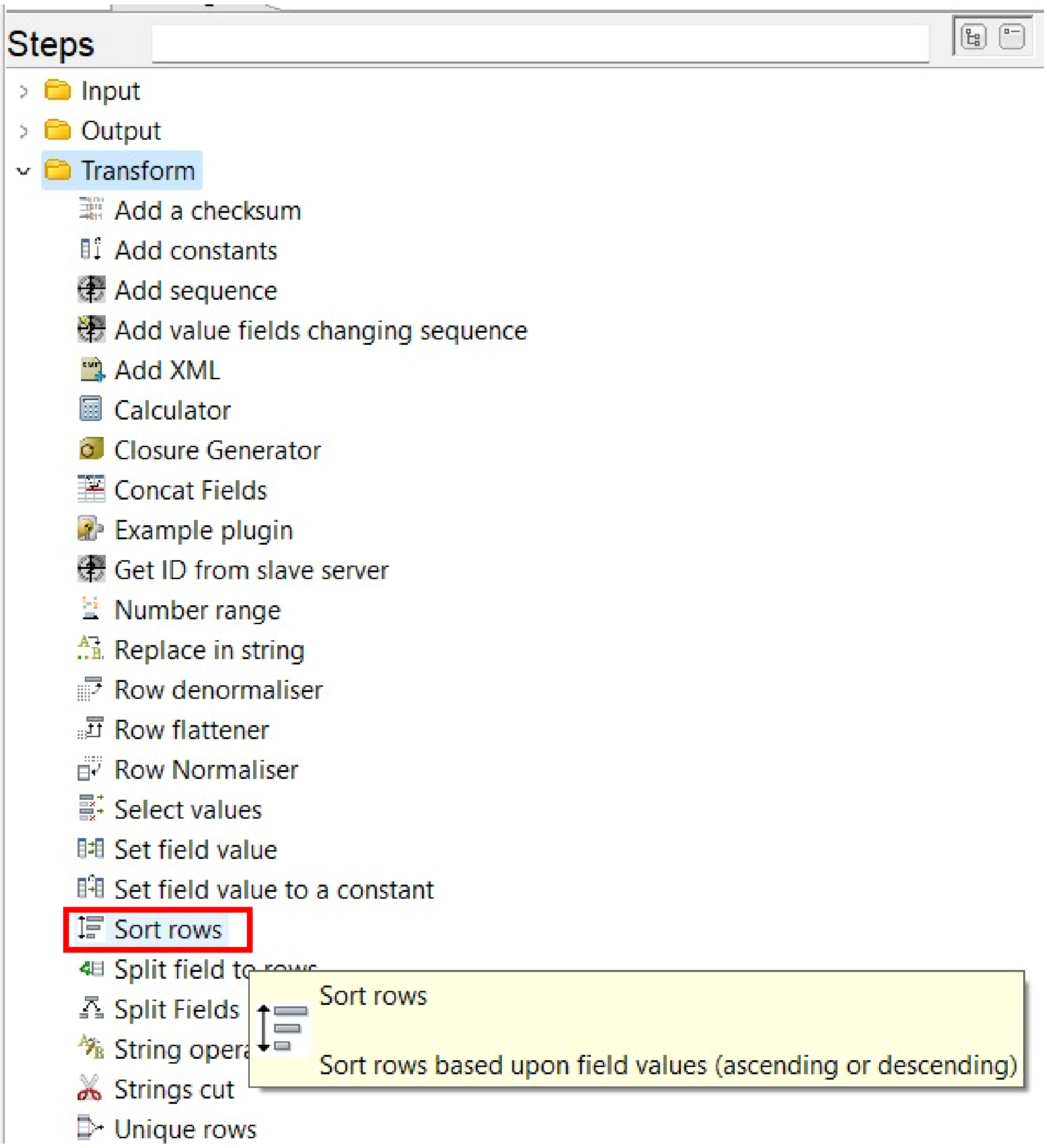


Sort rows

Sort rows based upon field values (ascending or descending)

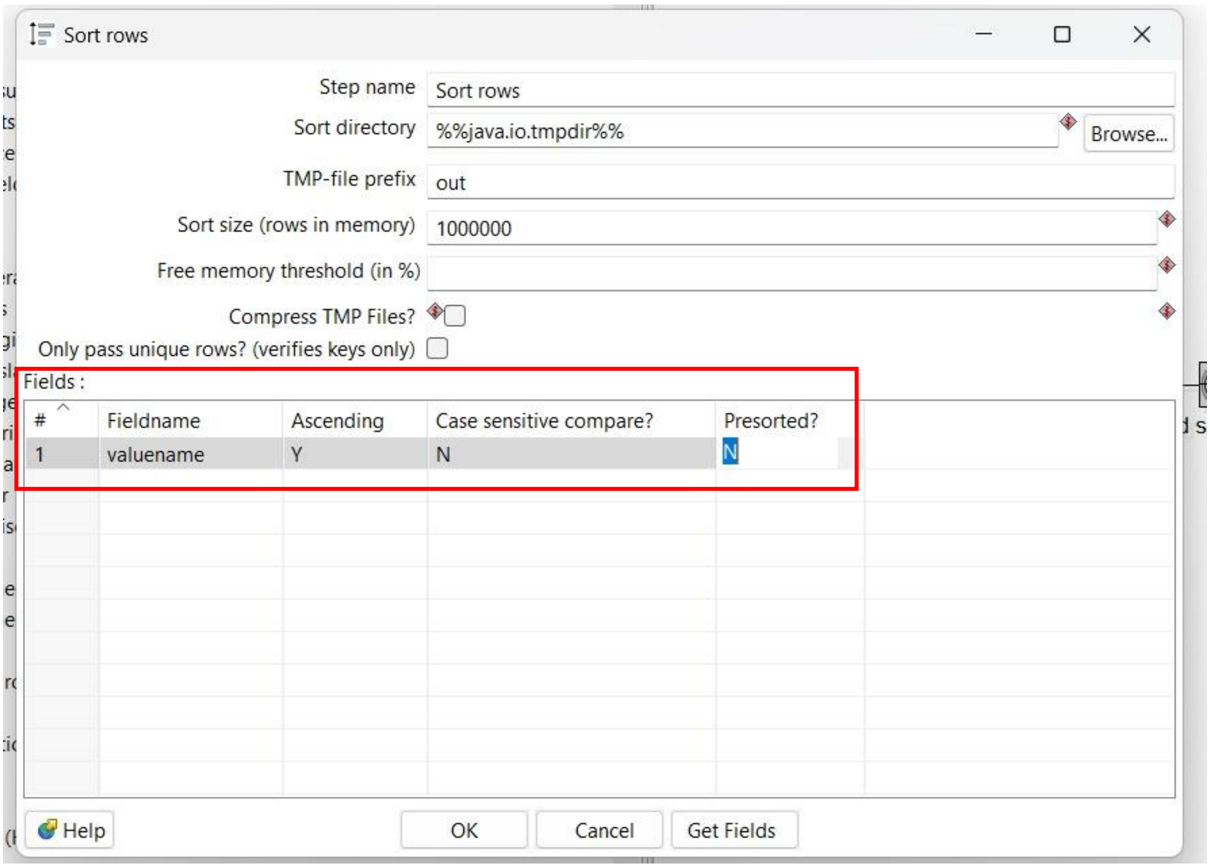
Hold the mouse pointer on Add sequence and select and drag the output connector to the Sort rows.





Sort rows

Sort rows based upon field values (ascending or descending)



Sort rows

Step name: Sort rows

Sort directory: %%java.io.tmpdir%%

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

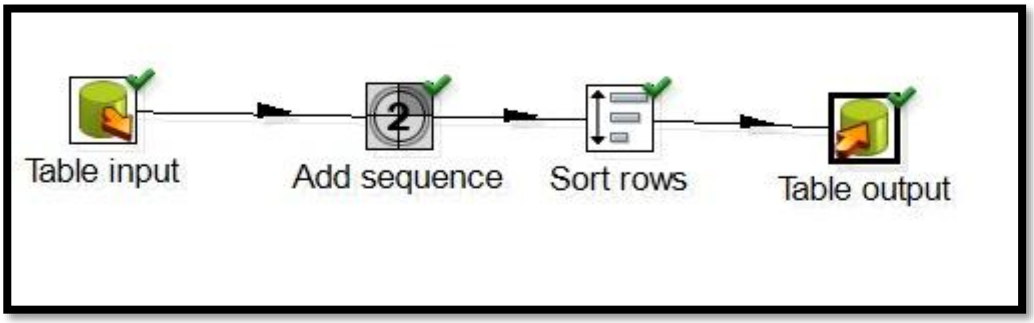
Compress TMP Files? □

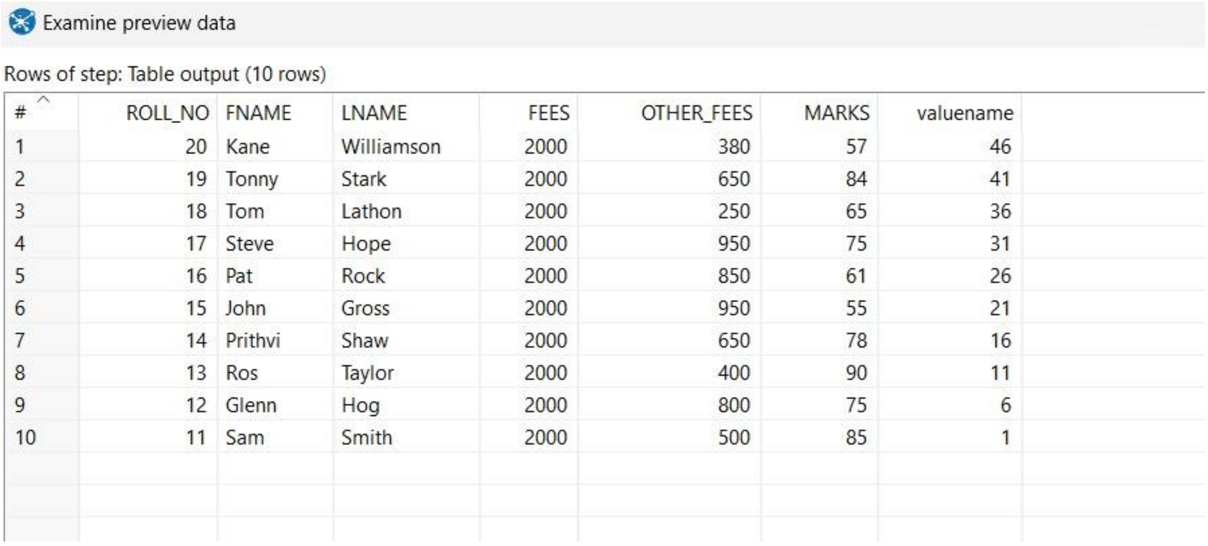
Only pass unique rows? (verifies keys only) □

Fields :

| # | Fieldname | Ascending | Case sensitive compare? | Presorted? |
|---|---|---|---|---|
| 1 | valuename | Y | N | N |

Step 4: Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





Examine preview data

Rows of step: Table output (10 rows)

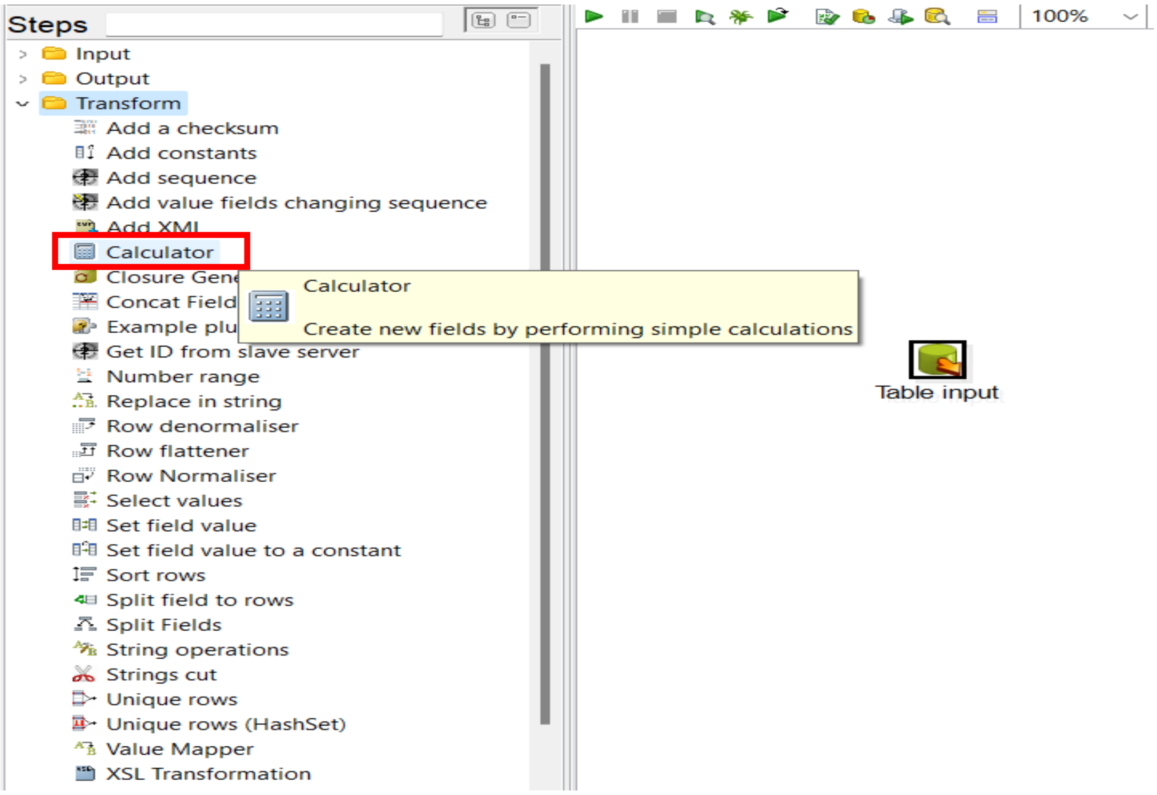
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | valuename |
|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | 46 |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | 41 |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | 36 |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | 31 |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | 26 |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | 21 |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | 16 |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | 11 |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | 6 |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | 1 |

```
SQL> select*from ADB11;

 ROLL_NO   FNAME    LNAME   FEES   OTHER_FEES   MARKS   VALUENAME
---------- -------- ------- ------ ------------ ------- -----------
    11      Sam      Smith    2000      500         85         1
    12      Glenn     Hog     2000      800         75         6
    13      Ros      Taylor   2000      400         90         11
    14      Prithvi   Shaw    2000      650         78         16
    15      John     Gross    2000      950         55         21
    16      Pat       Rock    2000      850         61         26
    17      Steve     Hope    2000      950         75         31
    18      Tom      Lathon   2000      250         65         36
    19      Tonny    Stark    2000      650         84         41
    20      Kane   Williamson 2000      380         57         46

10 rows selected.
```

## TRAMSFORMATION 3: - Calculator

Repeat Steps 1 to 3 from TRANSFORMATION 1.

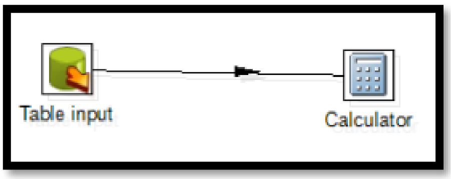
**Step 4: -** Perform Transformation Drag and drop Calculator from Transform folder under Design tab.



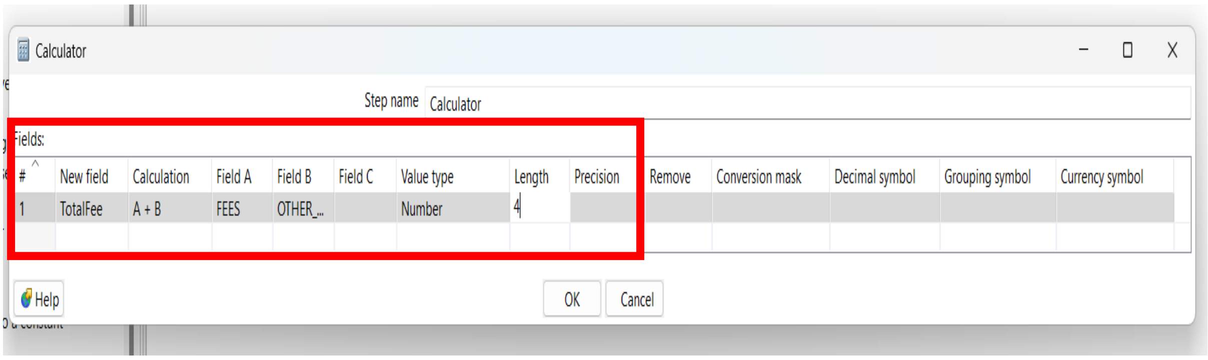
Calculator

Create new fields by performing simple calculations

Hold the mouse Pointer on Table input and select and drag the output connector to the Caculator.



Double Click on Calculator and fill in the details as shown below.



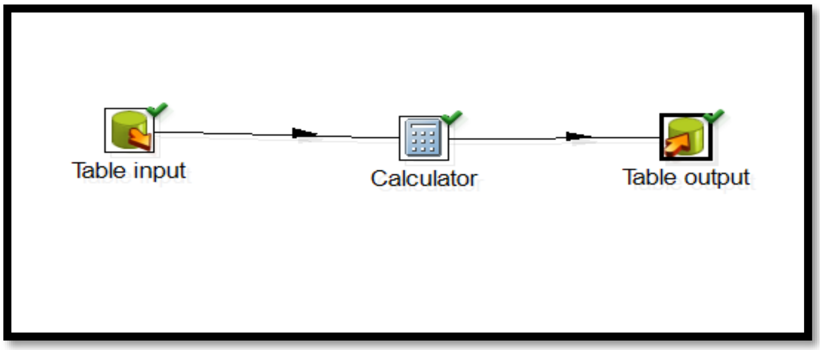
Step name Calculator

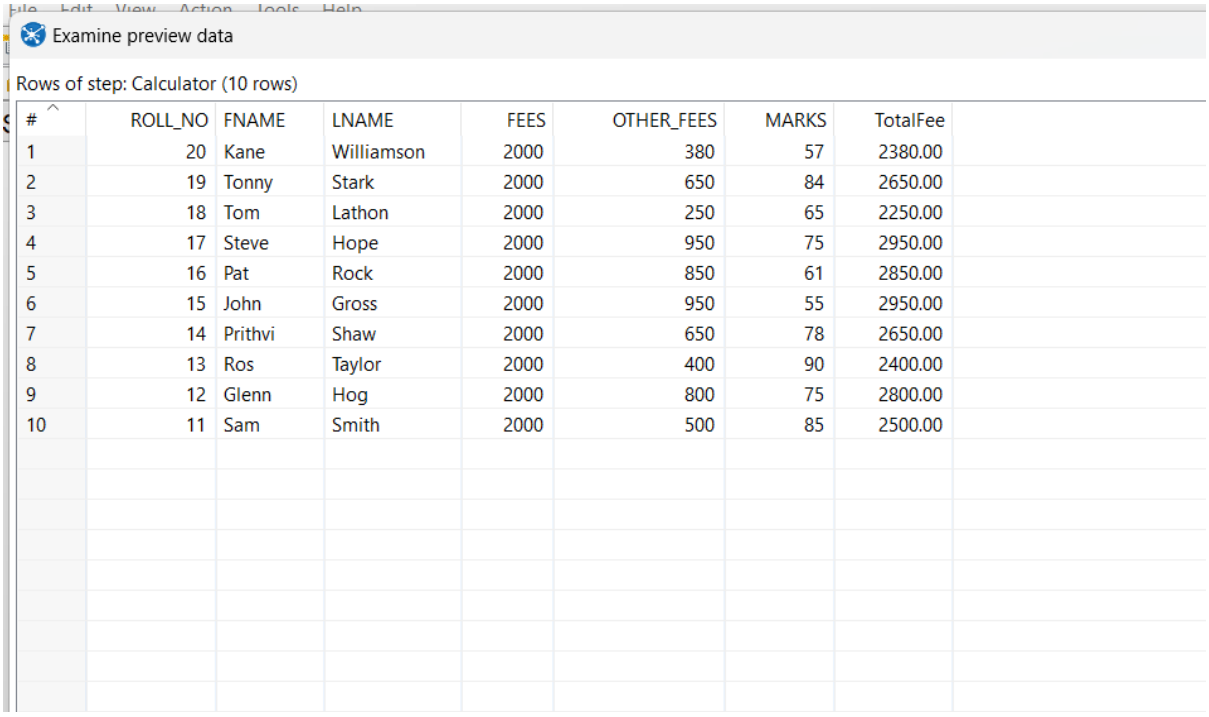
Fields:

| # | New field | Calculation | Field A | Field B | Field C | Value type | Length | Precision | Remove | Conversion mask | Decimal symbol | Grouping symbol | Currency symbol |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | TotalFee | A + B | FEES | OTHER_... | | Number | 4 | | | | | | |

This will add the values in Fees column & Other Fees column as result will be stored in TotalFee column. Click on OK.

**Step 5: -** Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





Examine preview data

Rows of step: Calculator (10 rows)

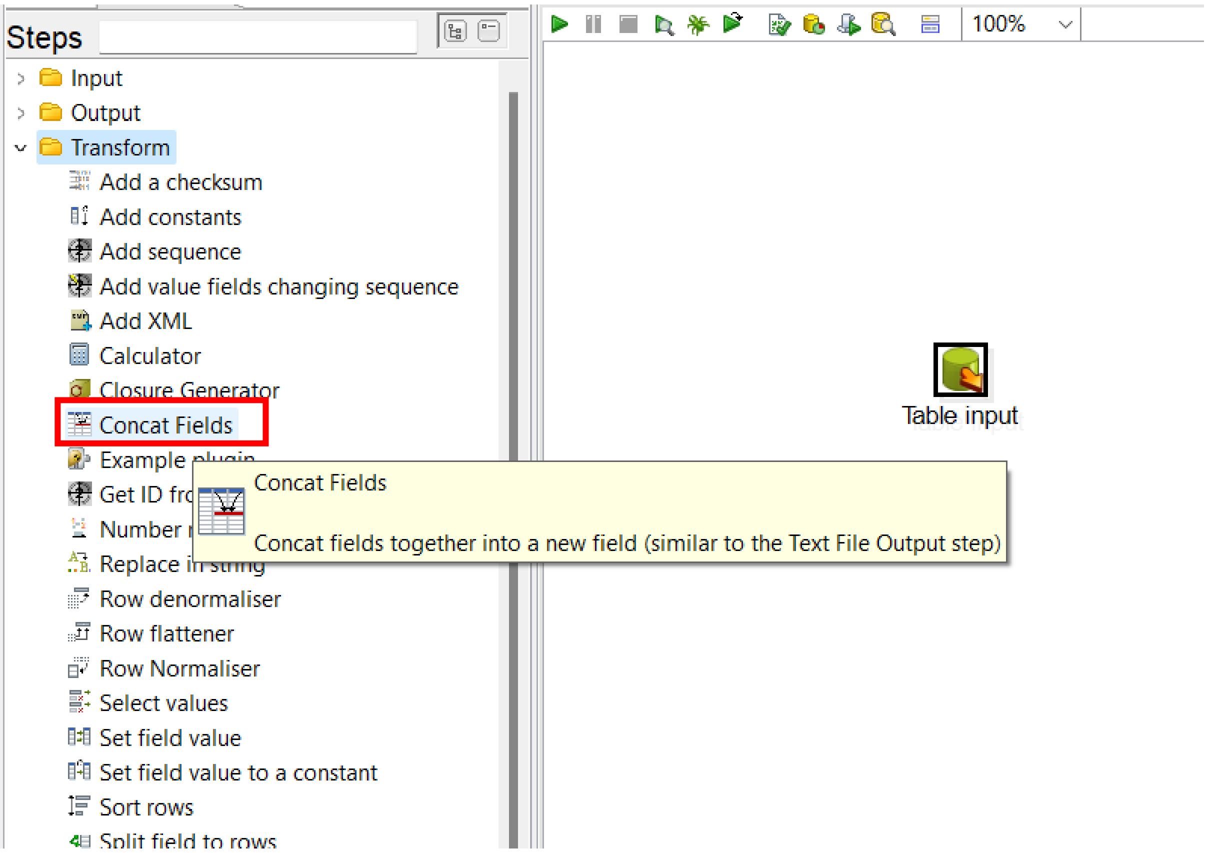
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | TotalFee |
|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | 2380.00 |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | 2650.00 |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | 2250.00 |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | 2950.00 |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | 2850.00 |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | 2950.00 |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | 2650.00 |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | 2400.00 |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | 2800.00 |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | 2500.00 |

```
SQL> select*from ADB2;

 ROLL_NO   FNAME    LNAME     FEES     OTHER_FEES    MARKS  TOTALFEE
---------------- -------------- -------------- ------------- ---------------------- ------------- --------------
   11       Sam      Smith     2000       500          85     2500
   12       Glenn    Hog       2000       800          75     2800
   13       Ros      Taylor    2000       400          90     2400
   14       Prithvi  Shaw      2000       650          78     2650
   15       John     Gross     2000       950          55     2950
   16       Pat      Rock      2000       850          61     2850
   17       Steve    Hope      2000       950          75     2950
   18       Tom      Lathon    2000       250          65     2250
   19       Tonny    Stark     2000       650          84     2650
   20       Kane   Williamson  2000       380          57     2380

10 rows selected.
```

**TRANSFORMATION 4: -** Concat Fields.

**Step 1: -** Repeat Steps 2 and 3 from TRANSFORMATION 1.

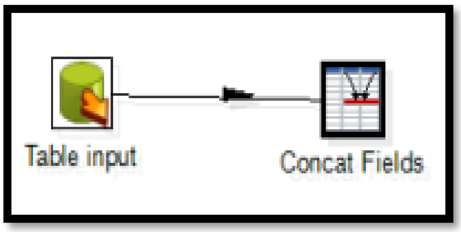
**Step 2: -** Perform Transformation. Drag and drop Concat Fields from Transform folder under Design tab.



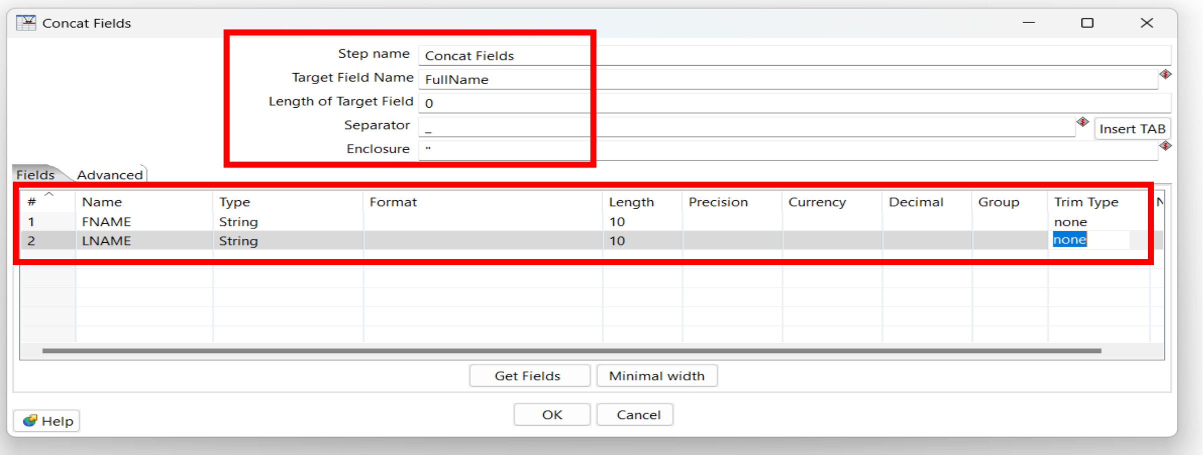
Concat Fields

Concat fields together into a new field (similar to the Text File Output step)

Hold the mouse Pointer on Table input and select and drag the output connector to the Concat Fields.



Double Click on Concat Fields and fill in the details as shown below→Click on OK.



Step name: Concat Fields

Target Field Name: FullName

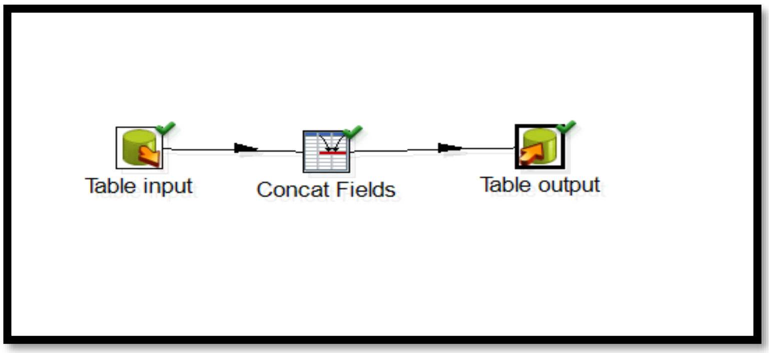
Length of Target Field: 0

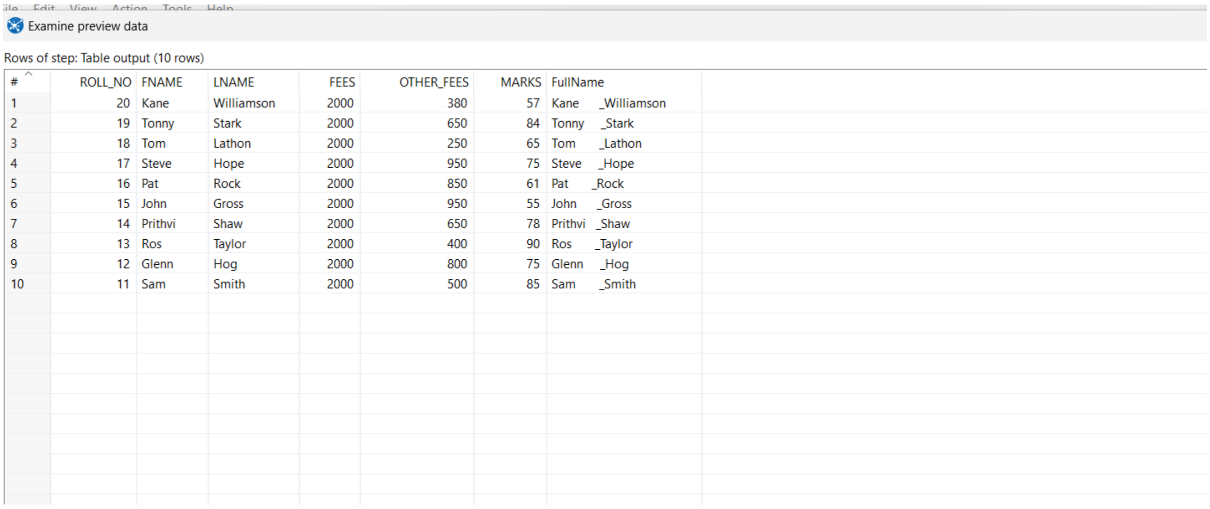
Separator: _

Enclosure: "

| # | Name | Type | Format | Length | Precision | Currency | Decimal | Group | Trim Type | N |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | FNAME | String | | 10 | | | | | none | |
| 2 | LNAME | String | | 10 | | | | | none | |

**Step 3: -** Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





Examine preview data

Rows of step: Table output (10 rows)

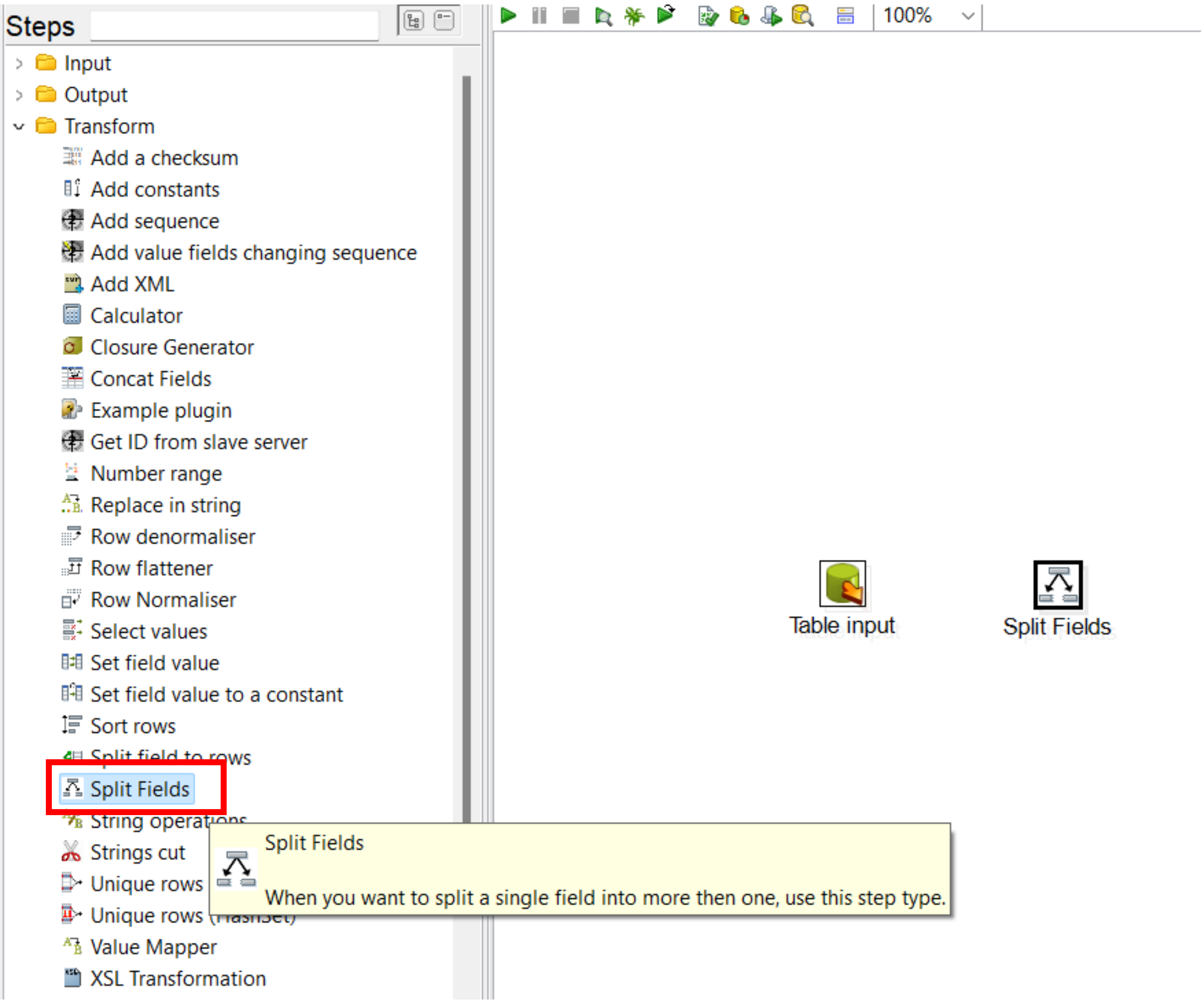
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | FullName |
|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | Kane _Williamson |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | Tonny _Stark |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | Tom _Lathon |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | Steve _Hope |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | Pat _Rock |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | John _Gross |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | Prithvi _Shaw |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | Ros _Taylor |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | Glenn _Hog |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | Sam _Smith |

```
SQL> select*from NEW;
 ROLL_NO   FNAME    LNAME    FEES    OTHER_FEES    MARKS    FULLNAME
--------------- ------------- -------------- ----------- --------------------- ---------------   ----------------

      11       Sam       Smith      2000        500           85       Sam _Smith


      12       Glenn      Hog       2000        800           75       Glenn _Hog


      13       Ros       Taylor     2000        400           90       Ros_Taylor


      14       Prithvi    Shaw      2000        650          78       Prithvi_Shaw


      15       John      Gross     2000        950           55       John_Gross


      16      Pat        Rock      2000        850           61       Pat_Rock


      17      Steve        Hope    2000        950           75      Steve_Hope


      18      Tom       Lathon     2000        250           65      Tom_Lathon


      19      Tonny       Stark    2000        650           84     Tonny_Stark


      20     Kane    Williamson   2000        380          57    Kane_Williamson

10 rows selected.
```

TRANSFORMATION 5: - Split Fields.

**Step 1: -** Repeat Steps 2 and 3 from TRANSFORMATION 1 (Import output table of concat fields transformation as Table input).

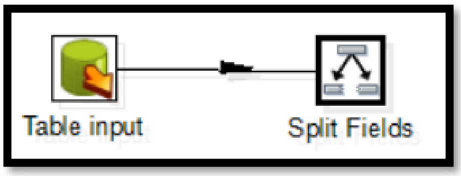
**Step 2: -** Perform Transformation. Drag and drop Concat Fields from Transform folder under Design tab.



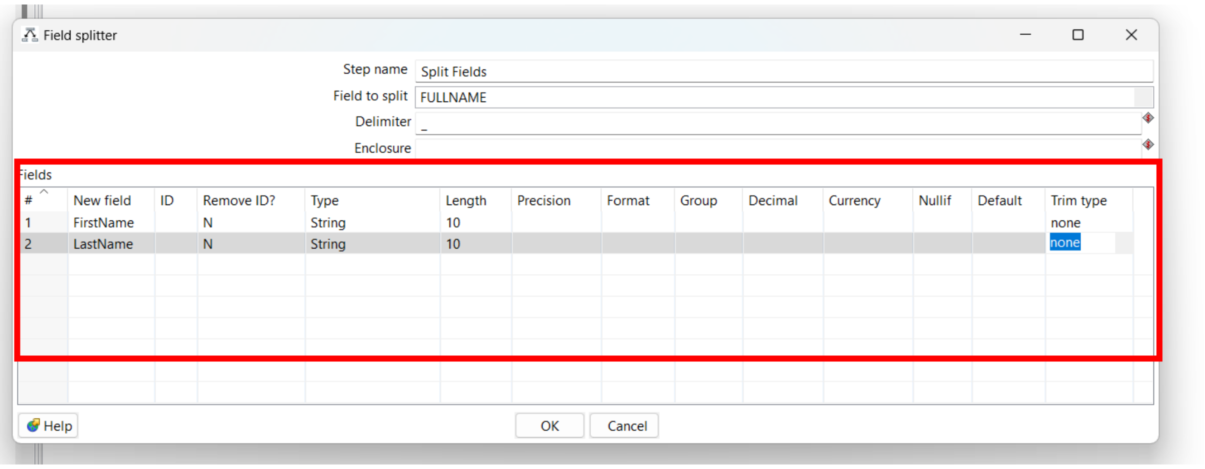
Split Fields

When you want to split a single field into more then one, use this step type.

Hold the mouse Pointer on Table input and select and drag the output connector to the Split Fields.



Double Click on Split Fields and fill in the details as shown below→Click on OK

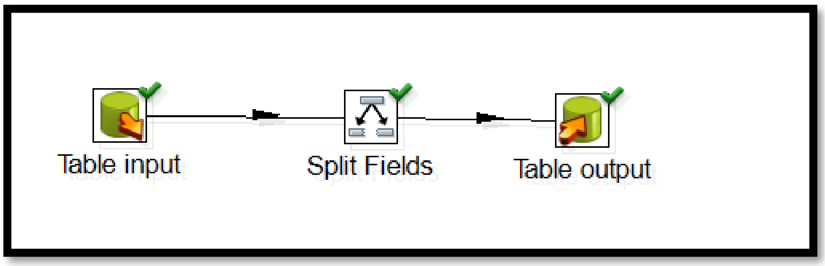


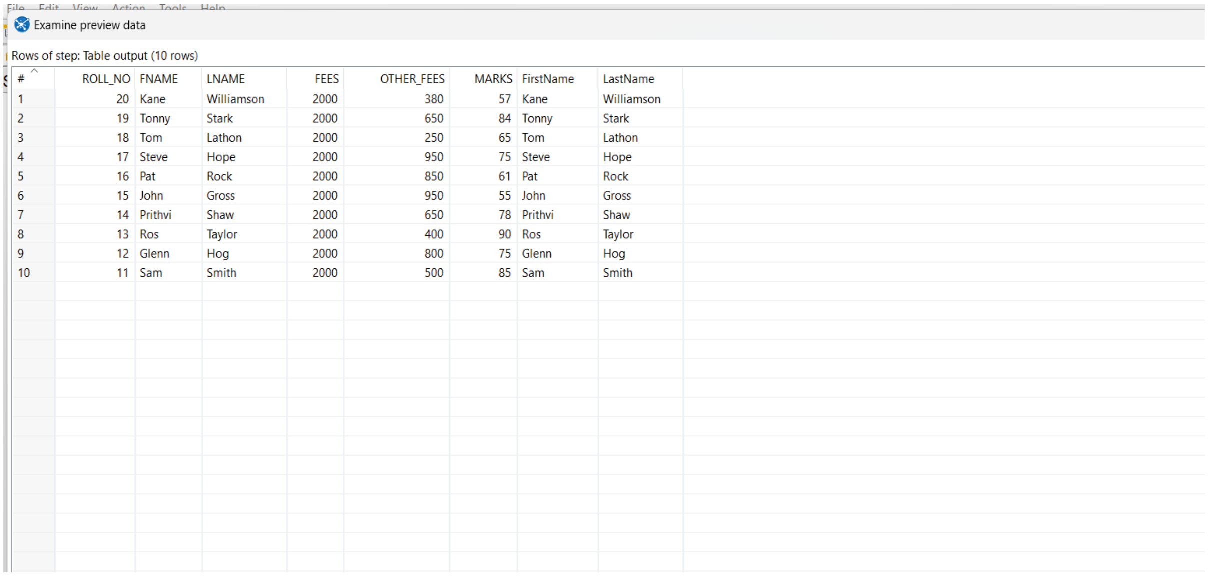
Step name: Split Fields
Field to split: FULLNAME
Delimiter: _
Enclosure:

Fields

| # | New field | ID | Remove ID? | Type | Length | Precision | Format | Group | Decimal | Currency | Nullif | Default | Trim type |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | FirstName | | N | String | 10 | | | | | | | | none |
| 2 | LastName | | N | String | 10 | | | | | | | | none |

**Step 3: -** Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





Rows of step: Table output (10 rows)

| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | FirstName | LastName |
|---|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | Kane | Williamson |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | Tonny | Stark |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | Tom | Lathon |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | Steve | Hope |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | Pat | Rock |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | John | Gross |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | Prithvi | Shaw |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | Ros | Taylor |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | Glenn | Hog |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | Sam | Smith |

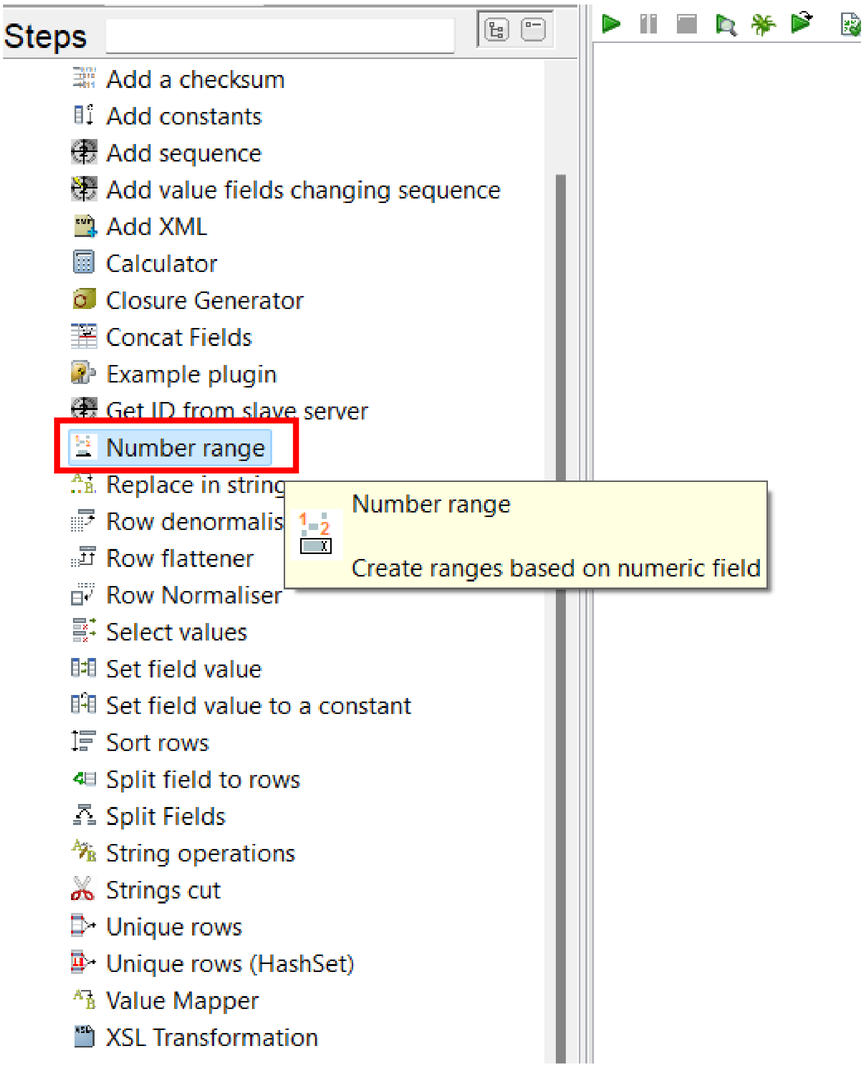
```
SQL> select*from New1;

  ROLL_NO  FNAME    LNAME    FEES    OTHER_FEES     MARKS  FIRSTNAME
---------------- ------------- -------------- --------- ----------------------- ------------- ------------------
LASTNAME
----------------------
      11          Sam          Smith       2000         500                      85            Sam
   Smith


      12         Glenn          Hog       2000         800                      75           Glenn
   Hog


      13          Ros           Taylor   2000         400                      90           Ros
   Taylor


      14         Prithvi         Shaw    2000          650                      78          Prithvi
   Shaw


      15         John            Gross   2000           950                      55          John
  Gross


      16          Pat            Rock   2000         850                       61          Pat
  Rock


      17        Steve            Hope  2000          950                        75         Steve
  Hope


      18       Tom              Lathon  2000         250                       65         Tom
 Lathon
```

```
ROLL_NO   FNAME    LNAME    FEES    OTHER_FEES     MARKS  FIRSTNAME
----------------- ------------- -------------- --------- ----------------------- ------------- ------------------
LASTNAME
----------------------


19              Tonny           Stark      2000        650                      84         Tonny
 Stark


 20             Kane        Williamson  2000        380                       57       Kane
 Williamson

10 rows selected.
```

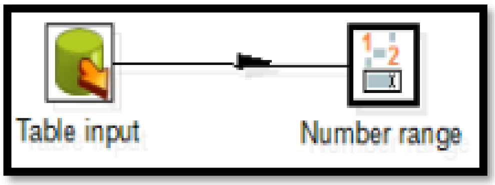
## TRANSFORMATION 6: Number Range

**Step 1: -** Repeat Steps 2 and 3 from TRANSFORMATION 1.

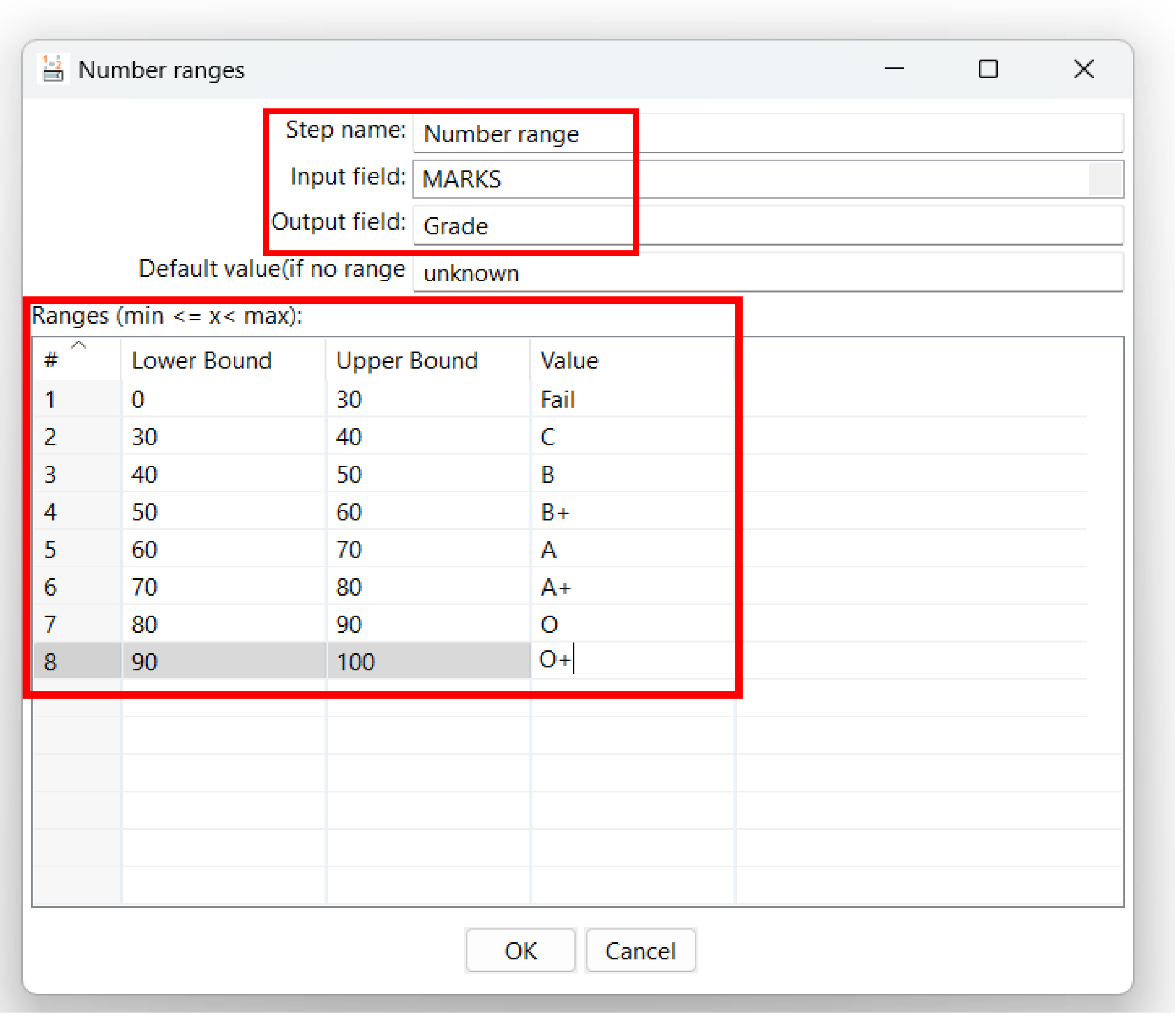
**Step 2: -** Perform Transformation. Drag and drop Number Range from Transform folder under Design tab.



Hold the mouse Pointer on Table input and select and drag the output connector to the Number range.



Double Click on Number range and fill in the details as shown below→Click on OK



Step name: Number range

Input field: MARKS

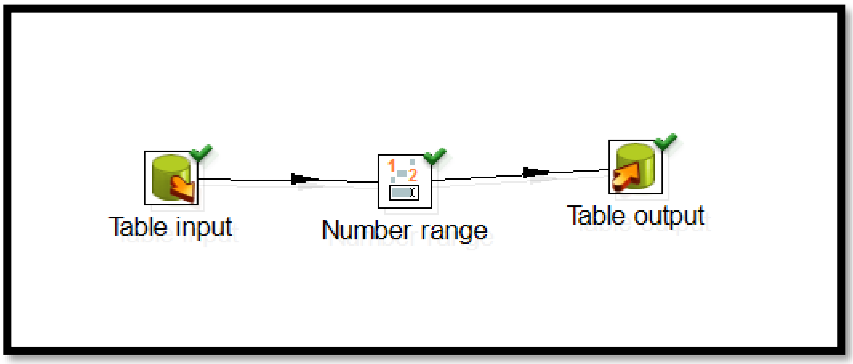
Output field: Grade

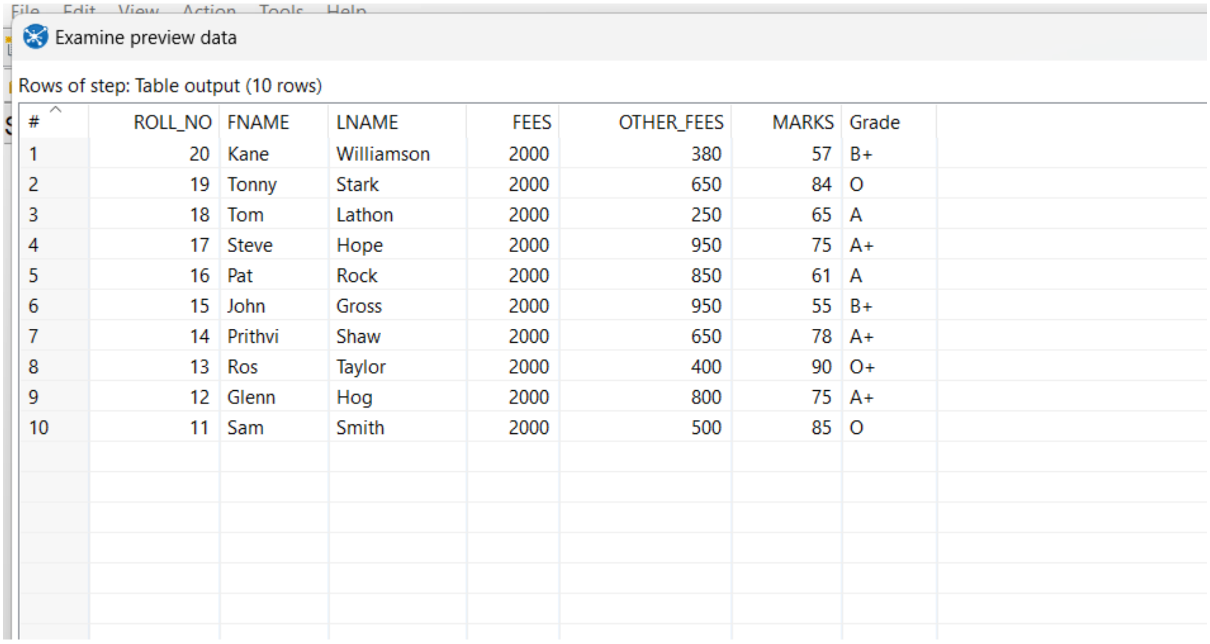
Default value(if no range unknown

Ranges (min <= x< max):

| # | Lower Bound | Upper Bound | Value |
|---|---|---|---|
| 1 | 0 | 30 | Fail |
| 2 | 30 | 40 | C |
| 3 | 40 | 50 | B |
| 4 | 50 | 60 | B+ |
| 5 | 60 | 70 | A |
| 6 | 70 | 80 | A+ |
| 7 | 80 | 90 | O |
| 8 | 90 | 100 | O+ |

**Step 3: -** Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





Examine preview data

Rows of step: Table output (10 rows)

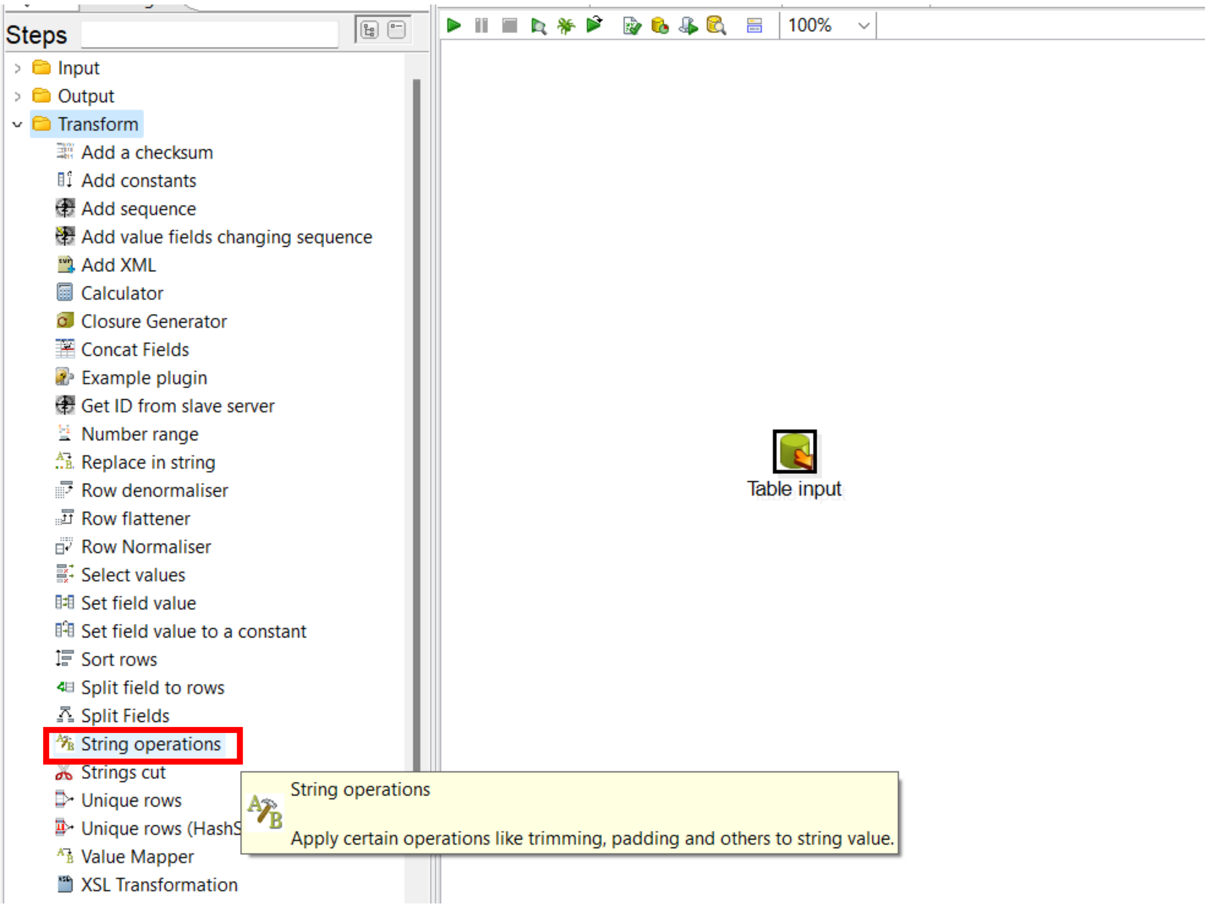
| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | Grade |
|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | B+ |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | O |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | A |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | A+ |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | A |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | B+ |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | A+ |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | O+ |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | A+ |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | O |

```
SQL> select*from ADB4;

  ROLL_NO    FNAME   LNAME    FEES   OTHER_FEES    MARKS      GRADE
----------------- ------------- -------------- --------- ---------------------- --------------- ----------------
     11          Sam       Smith      2000        500           85              O
     12         Glenn      Hog        2000        800           75              A+
     13          Ros       Taylor     2000        400           90              O+
     14        Prithvi     Shaw      2000        650           78             A+
     15         John       Gross     2000        950           55             B+
     16          Pat        Rock     2000        850           61              A
     17        Steve       Hope      2000        950           75              A+
     18         Tom       Lathon    2000        250           65             A
     19       Tonny       Stark     2000        650           84             O
     20         Kane    Williamson  2000        380           57             B+

10 rows selected.
```

## TRANSFORMATION 7: String Operations

**Step 1: -** Repeat Steps 2 and 3 from TRANSFORMATION 1.

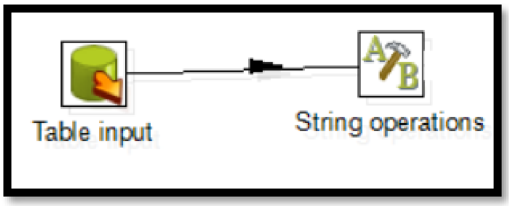
**Step 2: -** Perform Transformation. Drag and drop Number Range from Transform folder under Design tab.



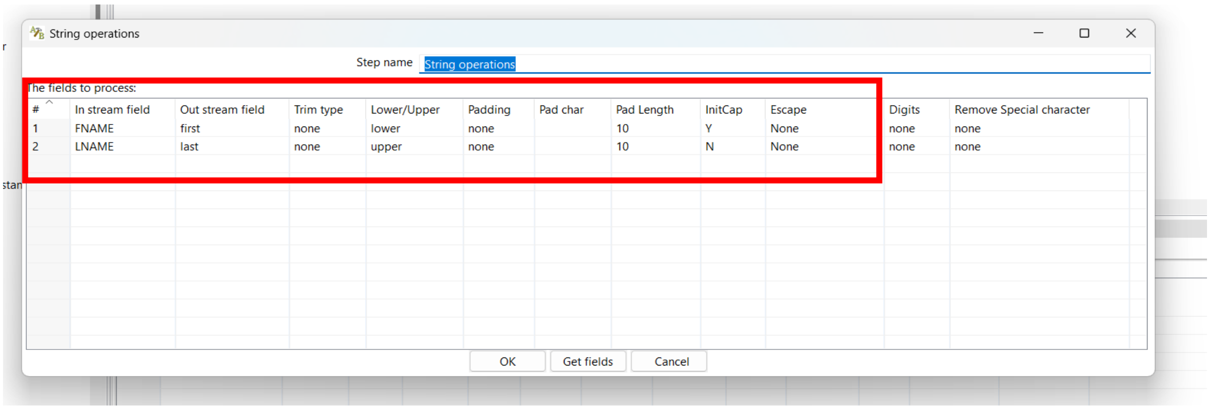
String operations

Apply certain operations like trimming, padding and others to string value.

Hold the mouse Pointer on Table input and select and drag the output connector to the String operations.



Double Click on String operations and fill in the details as shown below→Click on OK.

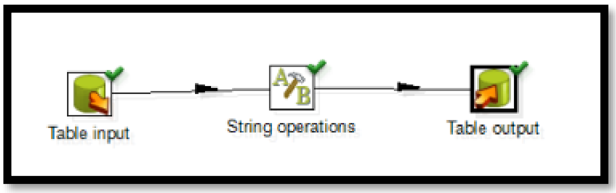


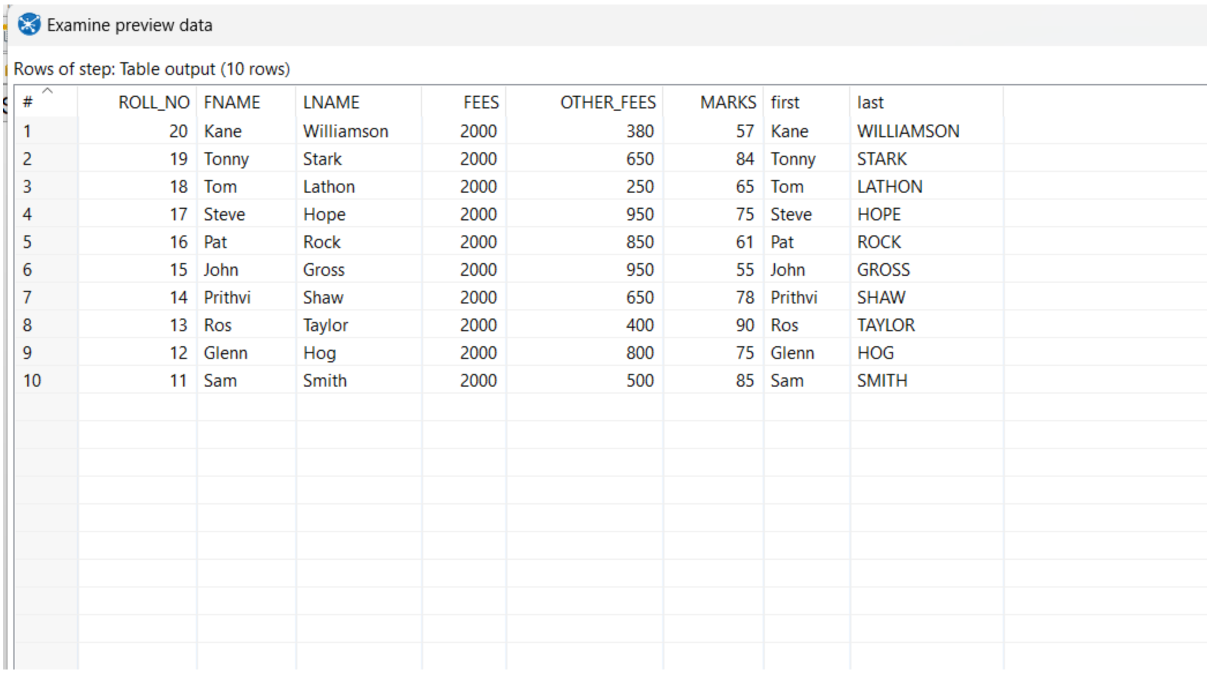
Step name: String operations

The fields to process:

| # | In stream field | Out stream field | Trim type | Lower/Upper | Padding | Pad char | Pad Length | InitCap | Escape | Digits | Remove Special character |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | FNAME | first | none | lower | none | | 10 | Y | None | none | none |
| 2 | LNAME | last | none | upper | none | | 10 | N | None | none | none |

Step 3: - Repeat Step 4 from TRANSFORMATION 1. If the Transformation is successful, you will see green ticks.





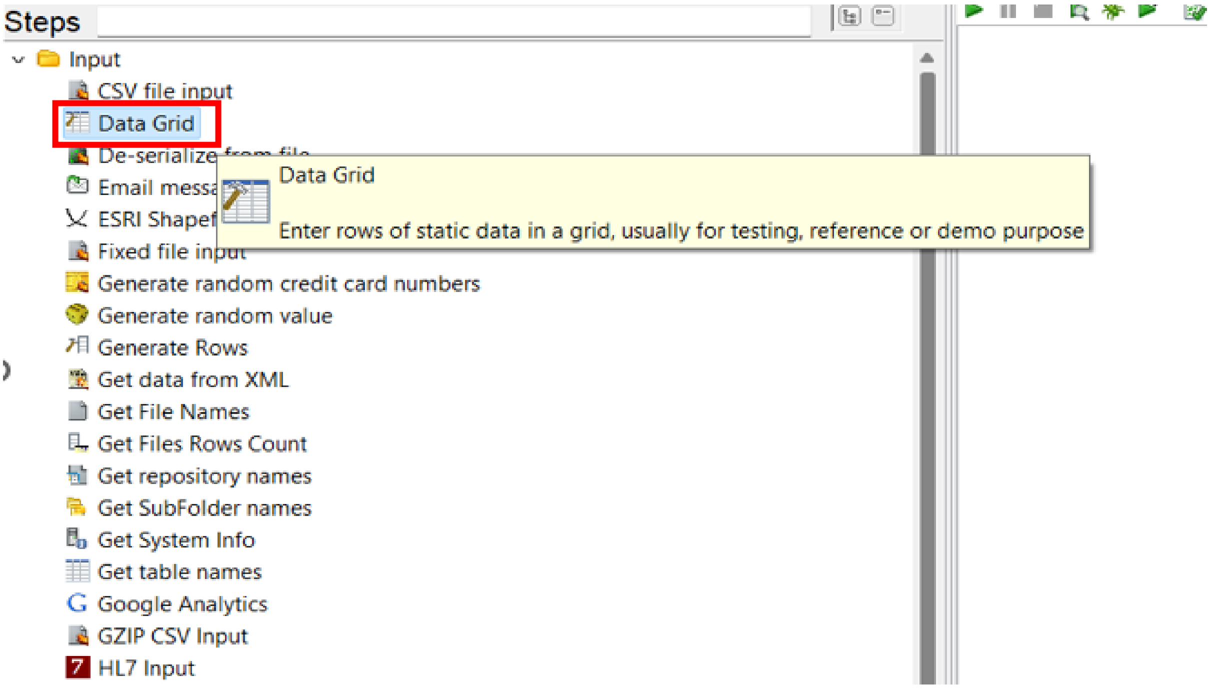
Rows of step: Table output (10 rows)

| # | ROLL_NO | FNAME | LNAME | FEES | OTHER_FEES | MARKS | first | last |
|---|---|---|---|---|---|---|---|---|
| 1 | 20 | Kane | Williamson | 2000 | 380 | 57 | Kane | WILLIAMSON |
| 2 | 19 | Tonny | Stark | 2000 | 650 | 84 | Tonny | STARK |
| 3 | 18 | Tom | Lathon | 2000 | 250 | 65 | Tom | LATHON |
| 4 | 17 | Steve | Hope | 2000 | 950 | 75 | Steve | HOPE |
| 5 | 16 | Pat | Rock | 2000 | 850 | 61 | Pat | ROCK |
| 6 | 15 | John | Gross | 2000 | 950 | 55 | John | GROSS |
| 7 | 14 | Prithvi | Shaw | 2000 | 650 | 78 | Prithvi | SHAW |
| 8 | 13 | Ros | Taylor | 2000 | 400 | 90 | Ros | TAYLOR |
| 9 | 12 | Glenn | Hog | 2000 | 800 | 75 | Glenn | HOG |
| 10 | 11 | Sam | Smith | 2000 | 500 | 85 | Sam | SMITH |

```
SQL> select*from ADB5;

ROLL_NO    FNAME     LNAME    FEES   OTHER_FEES     MARKS      FIRST
--------------- --------------- --------------- ----------- --------------------- ------------- ---------------
LAST
-----------
      11          Sam        Smith     2000           500            85        Sam
 SMITH
      12         Glenn       Hog       2000           800            75       Glenn
 HOG
      13         Ros         Taylor    2000           400            90         Ros
 TAYLOR
      14       Prithvi       Shaw     2000           650            78       Prithvi
SHAW
      15        John        Gross    2000           950            55       John
GROSS
      16         Pat        Rock     2000           850            61       Pat
ROCK
      17       Steve        Hope   2000            950            75       Steve
HOPE
      18        Tom       Lathon 2000            250            65      Tom
LATHON
      19       Tonny       Stark  2000           650            84     Tonny
 STARK
      20       Kane     Williamson 2000          380            57     Kane
 WILLIAMSON

10 rows selected.
```

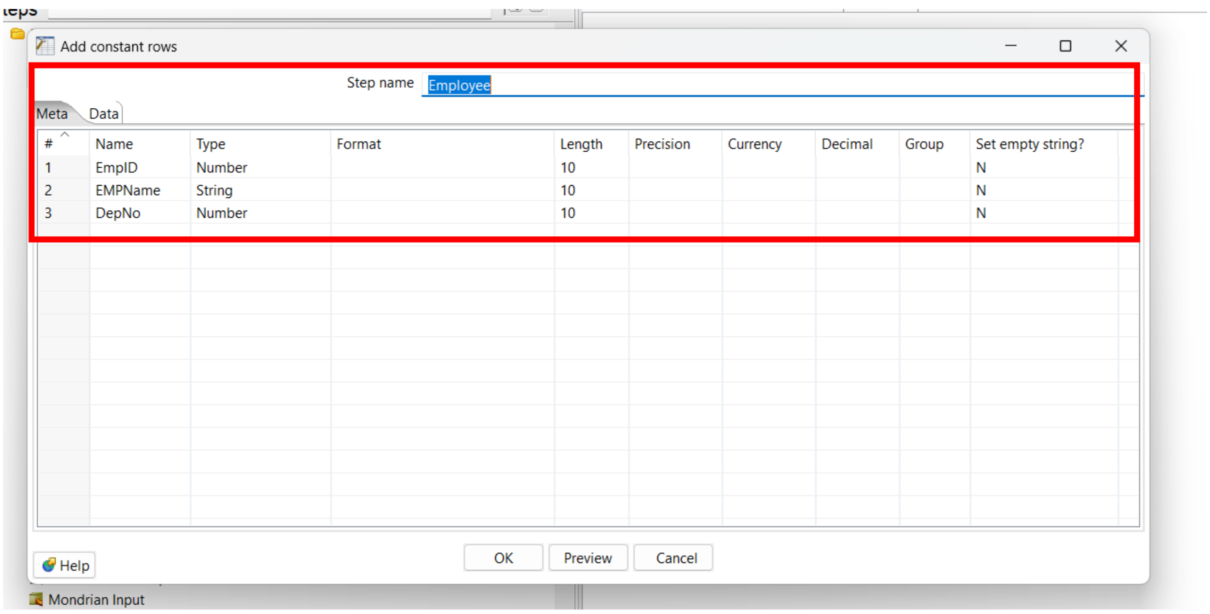
**TRANSFORMATION 8:** Merge Join

**Step 1: -** Drag and drop 2 Data Grid from Input folder under Design tab.



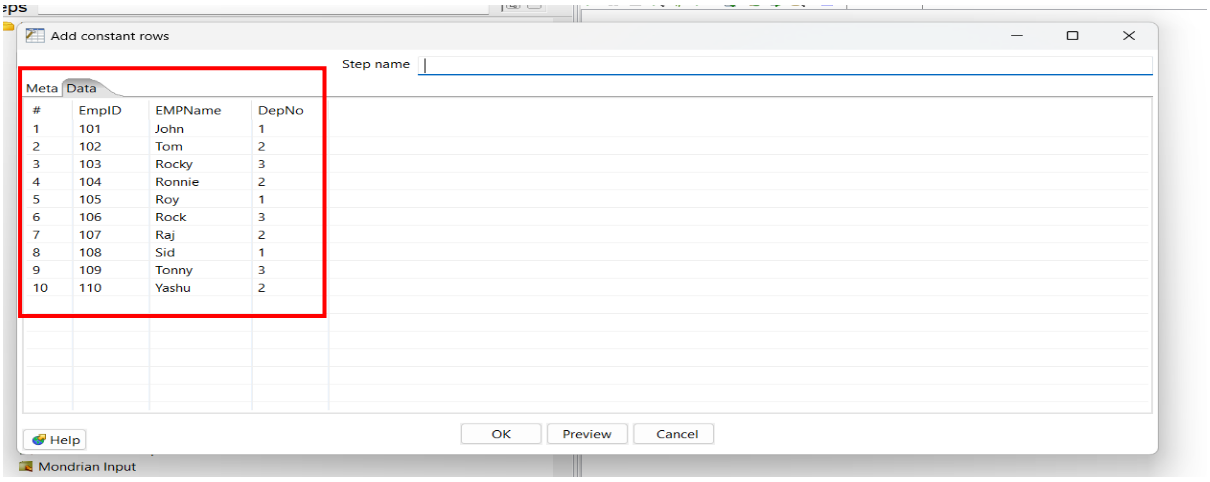
Rename them as Employee & Department.

**Step 2: -** Double click on them and insert records into respective grids→ Click on OK

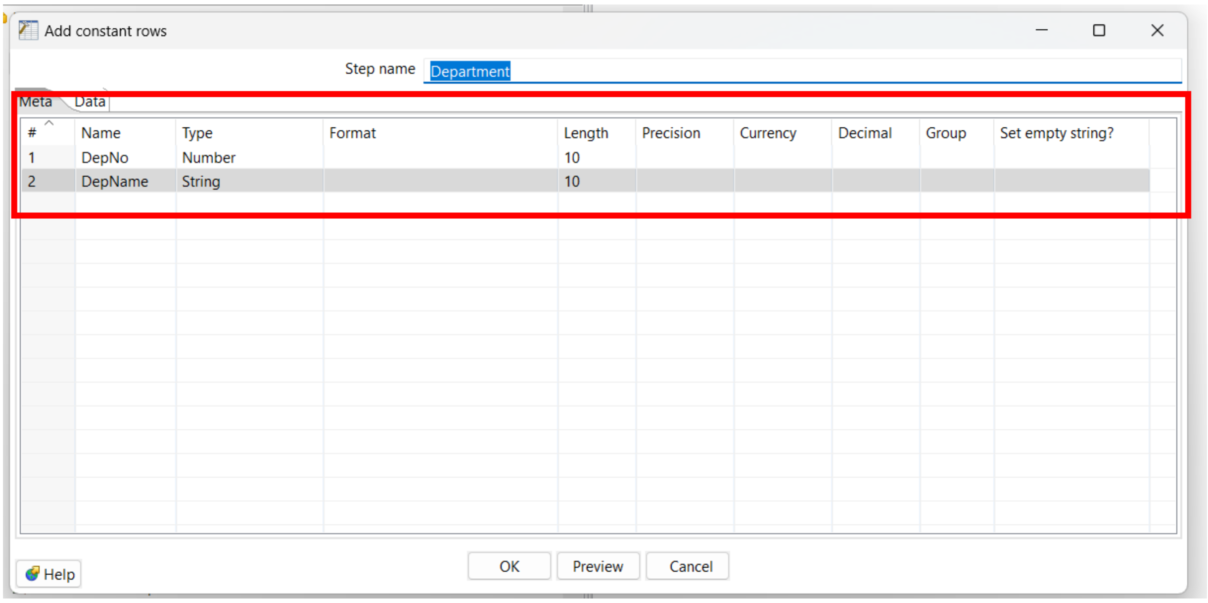


Step name: Employee

| # | Name | Type | Format | Length | Precision | Currency | Decimal | Group | Set empty string? |
|---|---|---|---|---|---|---|---|---|---|
| 1 | EmpID | Number | | 10 | | | | | N |
| 2 | EMPName | String | | 10 | | | | | N |
| 3 | DepNo | Number | | 10 | | | | | N |

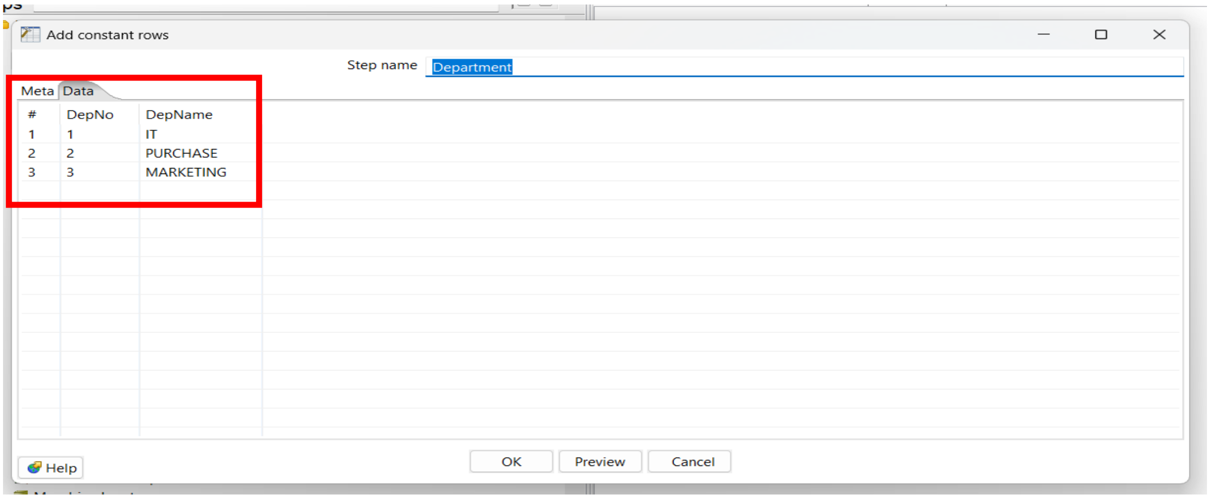


| # | EmpID | EMPName | DepNo |
|---|---|---|---|
| 1 | 101 | John | 1 |
| 2 | 102 | Tom | 2 |
| 3 | 103 | Rocky | 3 |
| 4 | 104 | Ronnie | 2 |
| 5 | 105 | Roy | 1 |
| 6 | 106 | Rock | 3 |
| 7 | 107 | Raj | 2 |
| 8 | 108 | Sid | 1 |
| 9 | 109 | Tonny | 3 |
| 10 | 110 | Yashu | 2 |



Step name Department

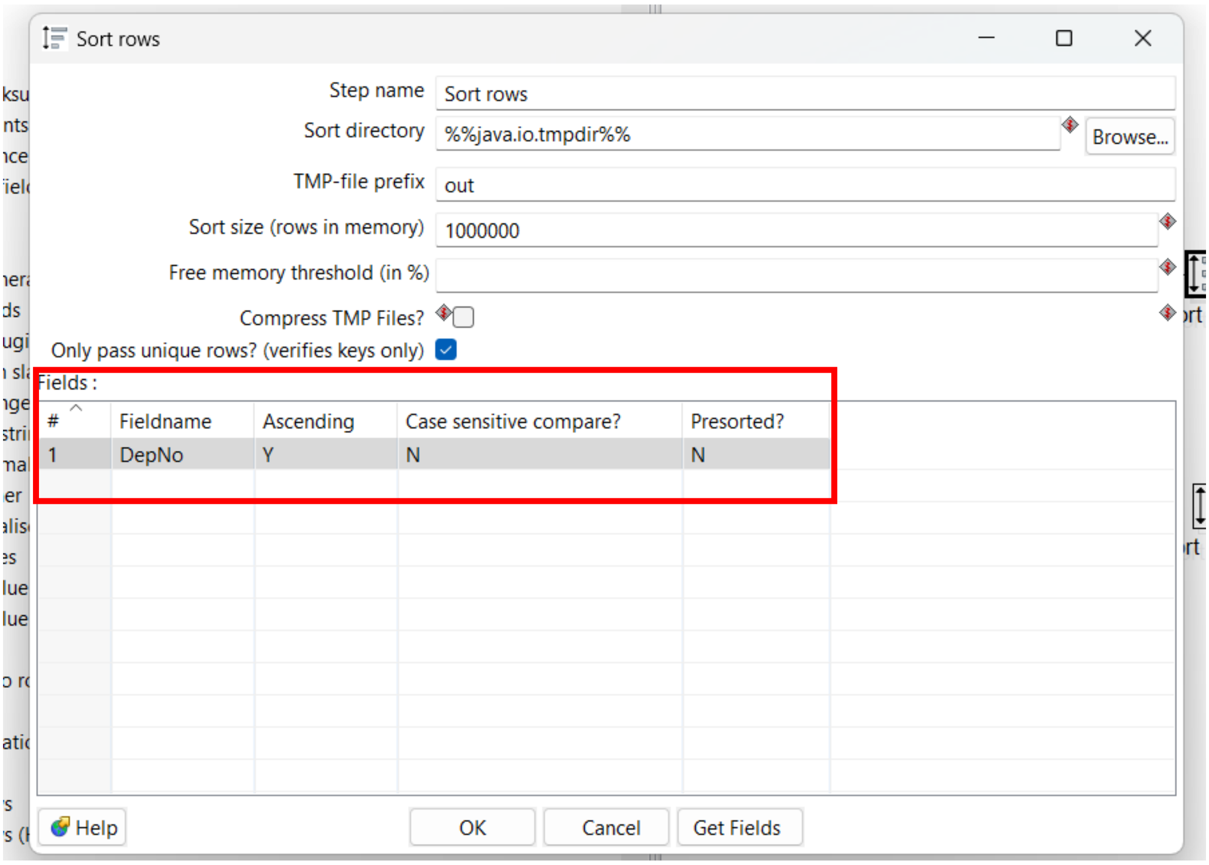
| # | Name | Type | Format | Length | Precision | Currency | Decimal | Group | Set empty string? |
|---|---|---|---|---|---|---|---|---|---|
| 1 | DepNo | Number | | 10 | | | | | |
| 2 | DepName | String | | 10 | | | | | |



Step name Department

| # | DepNo | DepName |
|---|---|---|
| 1 | 1 | IT |
| 2 | 2 | PURCHASE |
| 3 | 3 | MARKETING |

**Step 2: -** Perform Sort rows transformation for both data grids respectively. Click on OK.



Sort rows

Step name: Sort rows

Sort directory: %%java.io.tmpdir%%

TMP-file prefix: out

Sort size (rows in memory): 1000000

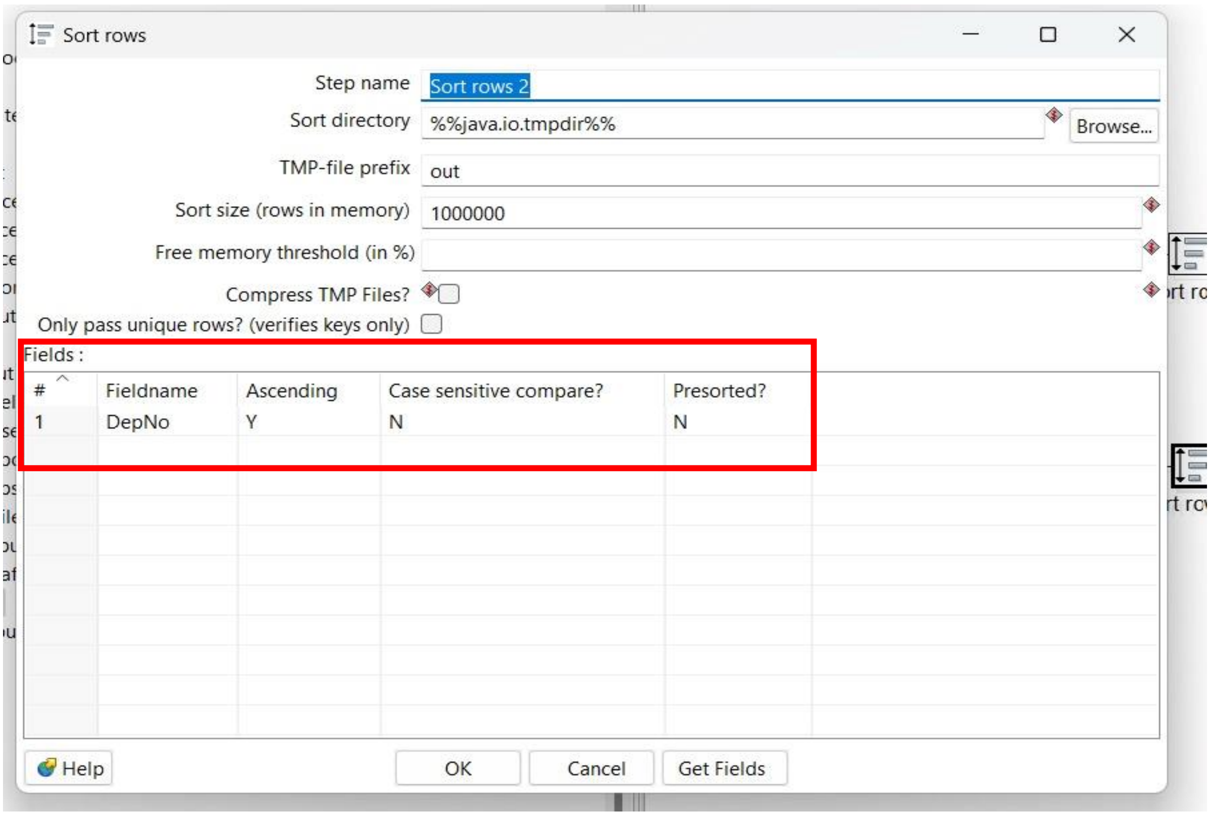
Free memory threshold (in %):

Compress TMP Files? □

Only pass unique rows? (verifies keys only) ☑

Fields :

| # | Fieldname | Ascending | Case sensitive compare? | Presorted? |
|---|---|---|---|---|
| 1 | DepNo | Y | N | N |



Sort rows

Step name: Sort rows 2

Sort directory: %%java.io.tmpdir%%

TMP-file prefix: out

Sort size (rows in memory): 1000000

Free memory threshold (in %):

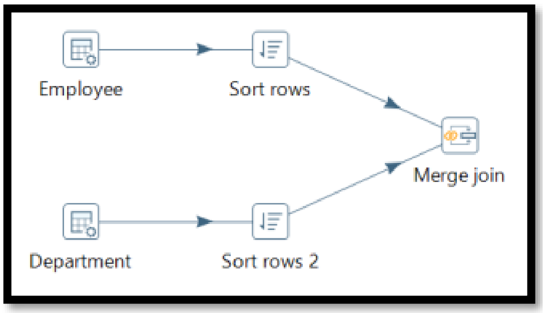
Compress TMP Files? □

Only pass unique rows? (verifies keys only) □

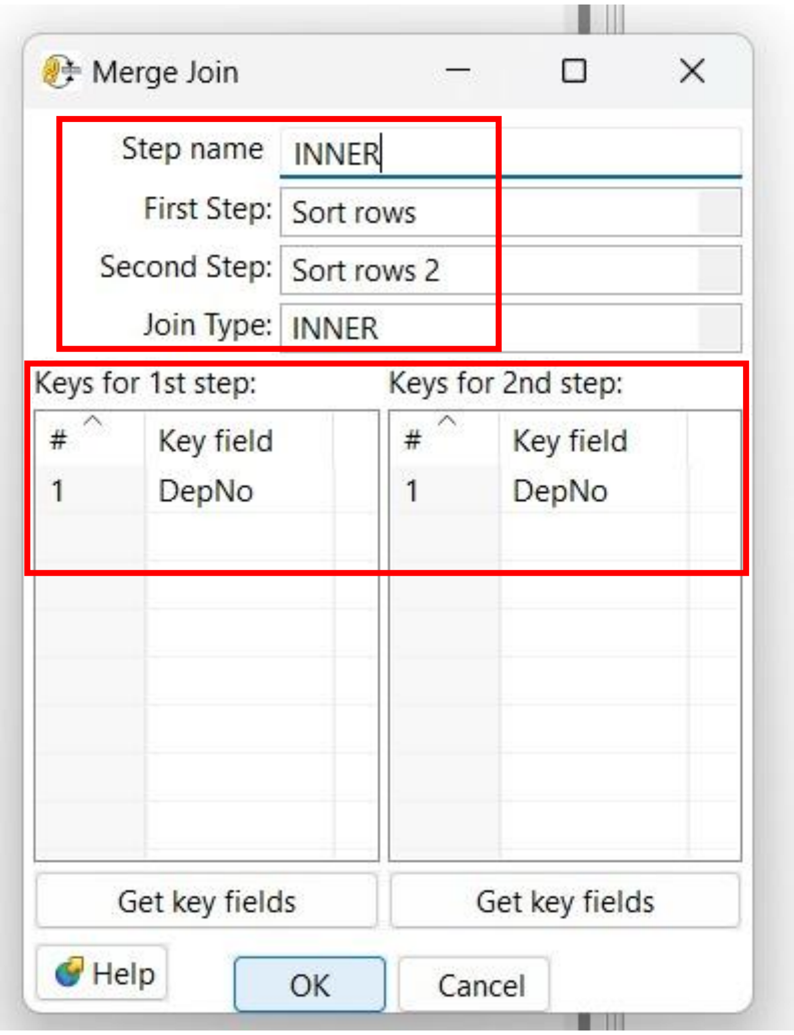
Fields :

| # | Fieldname | Ascending | Case sensitive compare? | Presorted? |
|---|---|---|---|---|
| 1 | DepNo | Y | N | N |

**Step 3: -** Drag and Drop Merge join from joins folder under Design tab. Hold the mouse Pointer on both the sort rows and select and drag the output connector to the Merge join as shown below.



**Step 4: -** Double click on Merge join and fill in the details as shown below to perform INNER join→Click on OK.



Step name: INNER

First Step: Sort rows

Second Step: Sort rows 2

Join Type: INNER

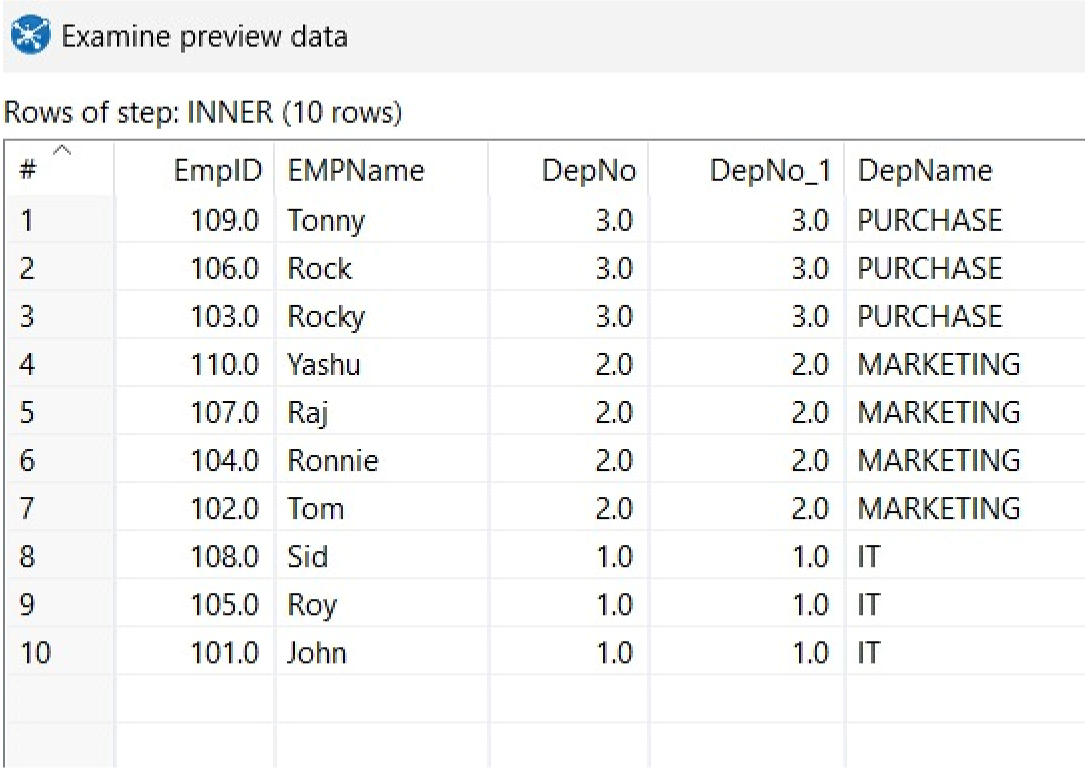
Keys for 1st step:

| # | Key field |
|---|---|
| 1 | DepNo |

Keys for 2nd step:

| # | Key field |
|---|---|
| 1 | DepNo |

Debug the transformation and perform Quick launch

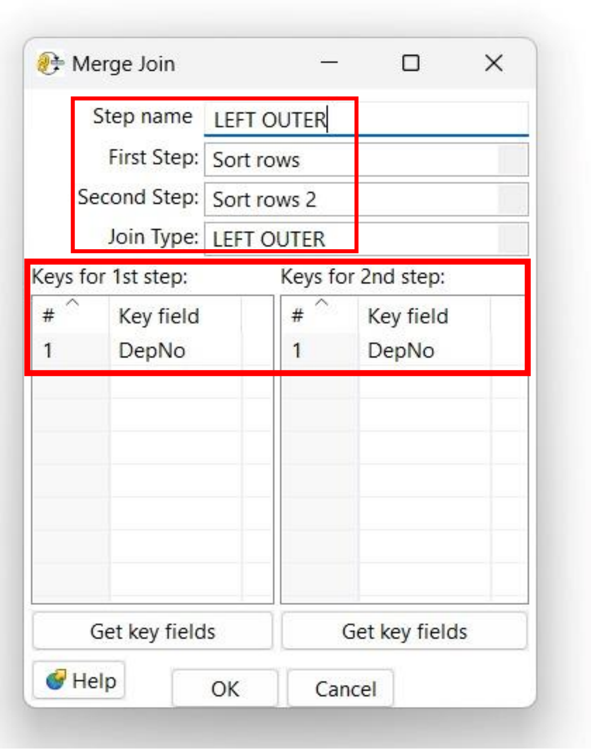


Examine preview data

Rows of step: INNER (10 rows)

| # | EmpID | EMPName | DepNo | DepNo_1 | DepName |
|---|---|---|---|---|---|
| 1 | 109.0 | Tonny | 3.0 | 3.0 | PURCHASE |
| 2 | 106.0 | Rock | 3.0 | 3.0 | PURCHASE |
| 3 | 103.0 | Rocky | 3.0 | 3.0 | PURCHASE |
| 4 | 110.0 | Yashu | 2.0 | 2.0 | MARKETING |
| 5 | 107.0 | Raj | 2.0 | 2.0 | MARKETING |
| 6 | 104.0 | Ronnie | 2.0 | 2.0 | MARKETING |
| 7 | 102.0 | Tom | 2.0 | 2.0 | MARKETING |
| 8 | 108.0 | Sid | 1.0 | 1.0 | IT |
| 9 | 105.0 | Roy | 1.0 | 1.0 | IT |
| 10 | 101.0 | John | 1.0 | 1.0 | IT |

**Step 5: -** Double click on Merge join and fill in the details as shown below to perform LEFT OUTER join→Click on OK.



Merge Join

Step name: LEFT OUTER

First Step: Sort rows

Second Step: Sort rows 2

Join Type: LEFT OUTER

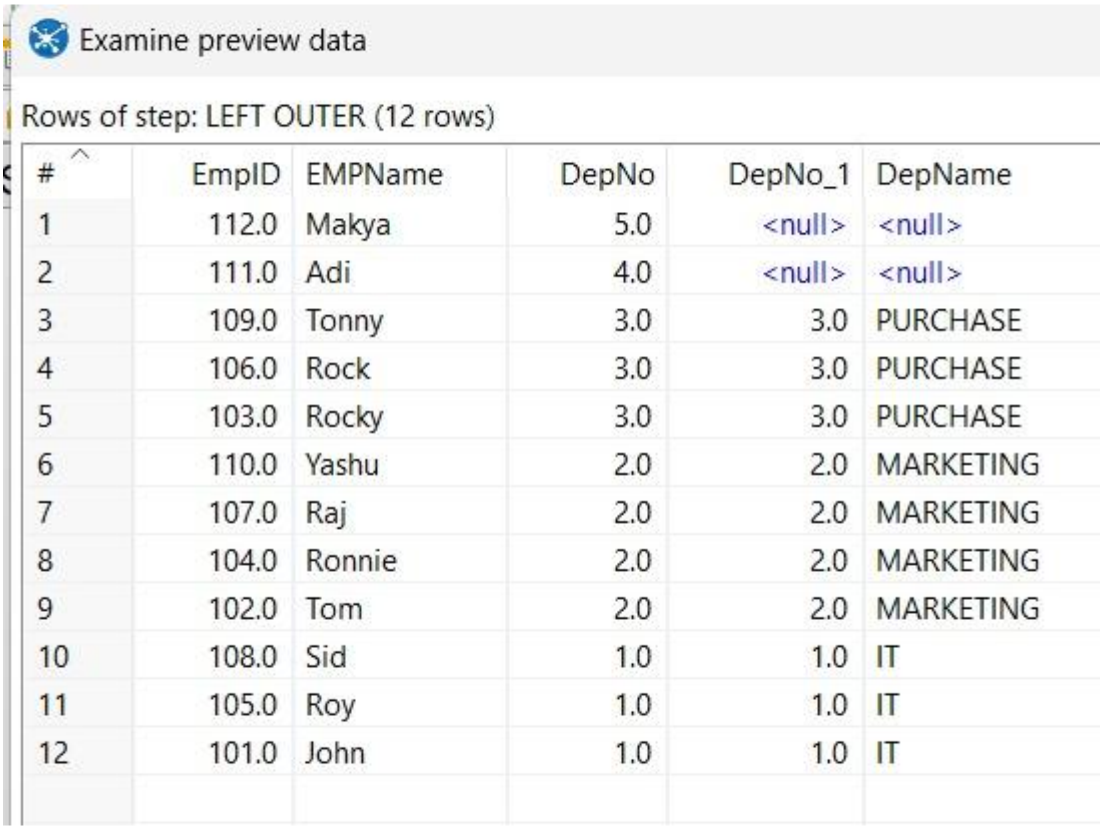
Keys for 1st step:

| # | Key field |
|---|---|
| 1 | DepNo |

Keys for 2nd step:

| # | Key field |
|---|---|
| 1 | DepNo |

Debug the transformation and perform Quick launch

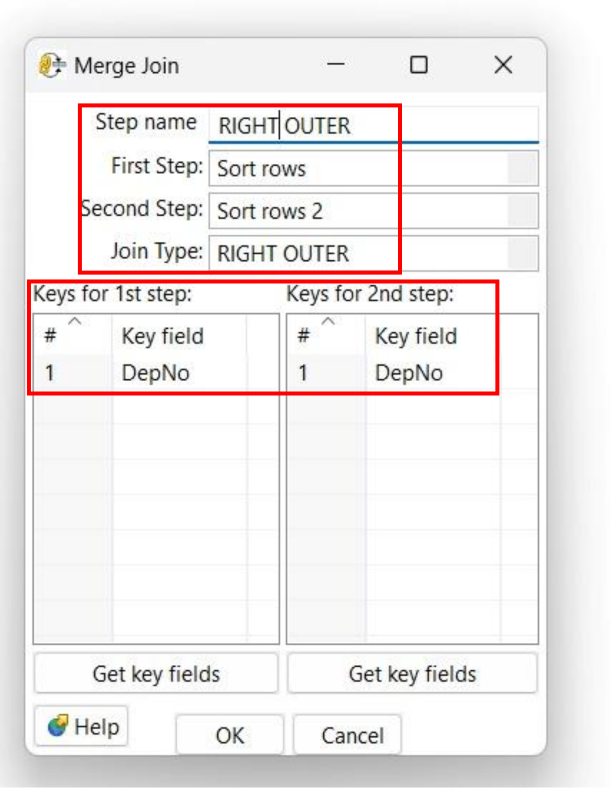


Examine preview data

Rows of step: LEFT OUTER (12 rows)

| # | EmpID | EMPName | DepNo | DepNo_1 | DepName |
|---|---|---|---|---|---|
| 1 | 112.0 | Makya | 5.0 | <null> | <null> |
| 2 | 111.0 | Adi | 4.0 | <null> | <null> |
| 3 | 109.0 | Tonny | 3.0 | 3.0 | PURCHASE |
| 4 | 106.0 | Rock | 3.0 | 3.0 | PURCHASE |
| 5 | 103.0 | Rocky | 3.0 | 3.0 | PURCHASE |
| 6 | 110.0 | Yashu | 2.0 | 2.0 | MARKETING |
| 7 | 107.0 | Raj | 2.0 | 2.0 | MARKETING |
| 8 | 104.0 | Ronnie | 2.0 | 2.0 | MARKETING |
| 9 | 102.0 | Tom | 2.0 | 2.0 | MARKETING |
| 10 | 108.0 | Sid | 1.0 | 1.0 | IT |
| 11 | 105.0 | Roy | 1.0 | 1.0 | IT |
| 12 | 101.0 | John | 1.0 | 1.0 | IT |

**Step 6: -** Double click on Merge join and fill in the details as shown below to perform RIGHT OUTER join→Click on OK.



Merge Join

Step name: RIGHT OUTER

First Step: Sort rows

Second Step: Sort rows 2

Join Type: RIGHT OUTER

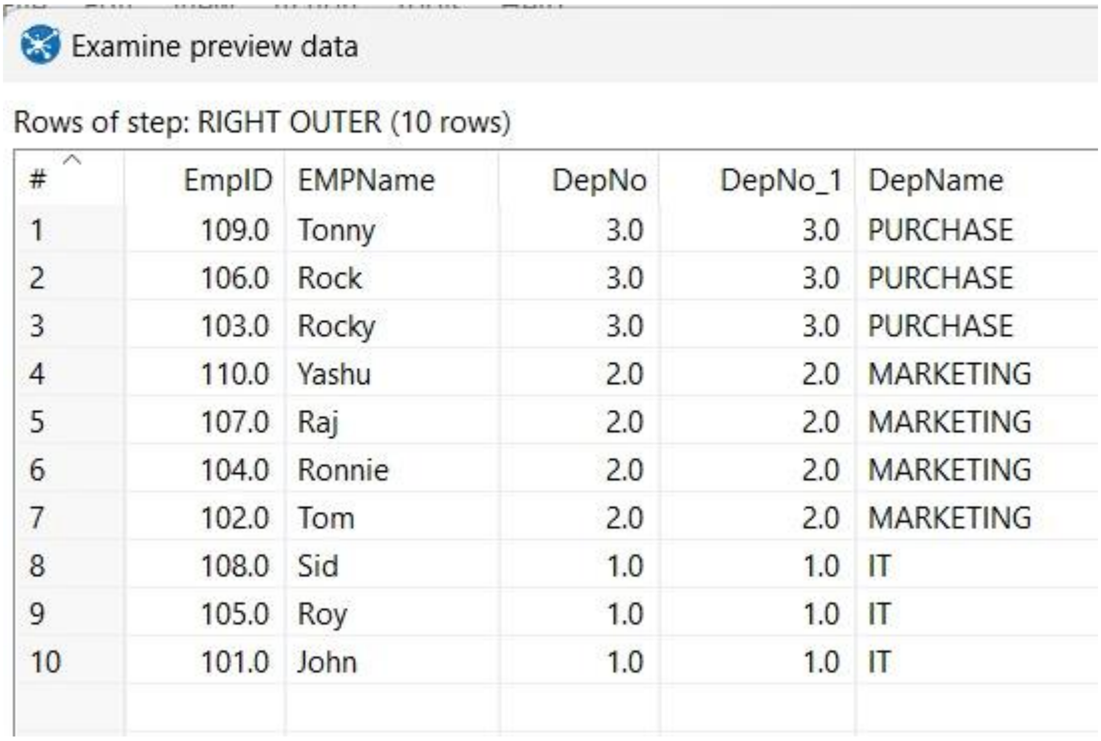
Keys for 1st step:

| # | Key field |
|---|---|
| 1 | DepNo |

Keys for 2nd step:

| # | Key field |
|---|---|
| 1 | DepNo |

Debug the transformation and perform Quick launch

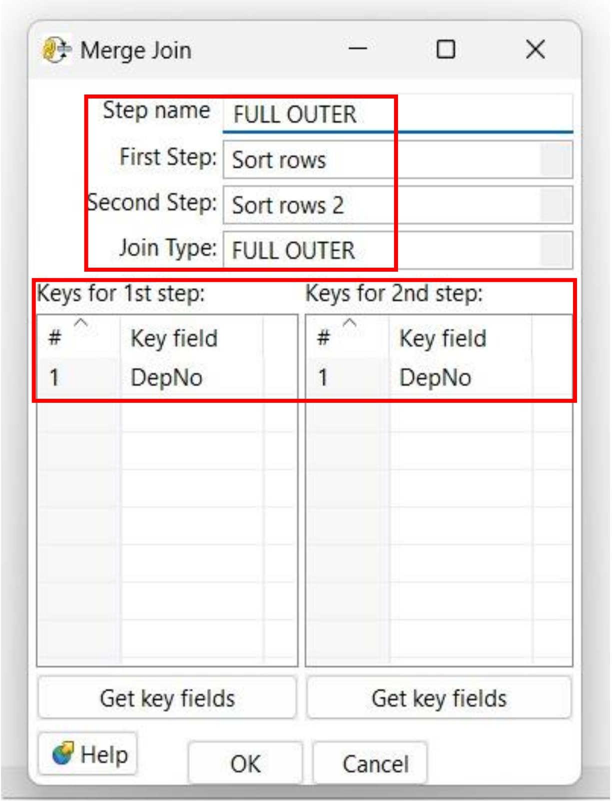


Examine preview data

Rows of step: RIGHT OUTER (10 rows)

| # | EmpID | EMPName | DepNo | DepNo_1 | DepName |
|---|---|---|---|---|---|
| 1 | 109.0 | Tonny | 3.0 | 3.0 | PURCHASE |
| 2 | 106.0 | Rock | 3.0 | 3.0 | PURCHASE |
| 3 | 103.0 | Rocky | 3.0 | 3.0 | PURCHASE |
| 4 | 110.0 | Yashu | 2.0 | 2.0 | MARKETING |
| 5 | 107.0 | Raj | 2.0 | 2.0 | MARKETING |
| 6 | 104.0 | Ronnie | 2.0 | 2.0 | MARKETING |
| 7 | 102.0 | Tom | 2.0 | 2.0 | MARKETING |
| 8 | 108.0 | Sid | 1.0 | 1.0 | IT |
| 9 | 105.0 | Roy | 1.0 | 1.0 | IT |
| 10 | 101.0 | John | 1.0 | 1.0 | IT |

**Step 6: -** Double click on Merge join and fill in the details as shown below to perform FULL OUTER join→Click on OK



Merge Join

Step name: FULL OUTER
First Step: Sort rows
Second Step: Sort rows 2
Join Type: FULL OUTER

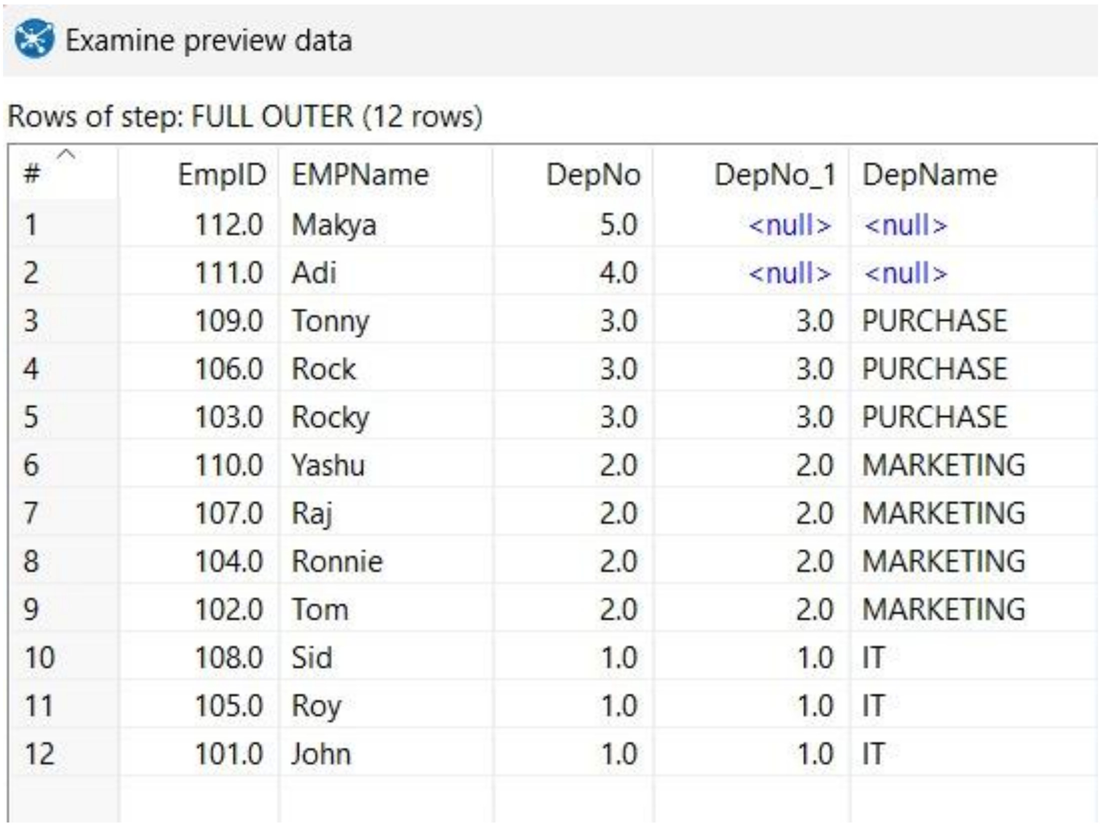
Keys for 1st step:

| # | Key field |
|---|---|
| 1 | DepNo |

Keys for 2nd step:

| # | Key field |
|---|---|
| 1 | DepNo |

Debug the transformation and perform Quick launch



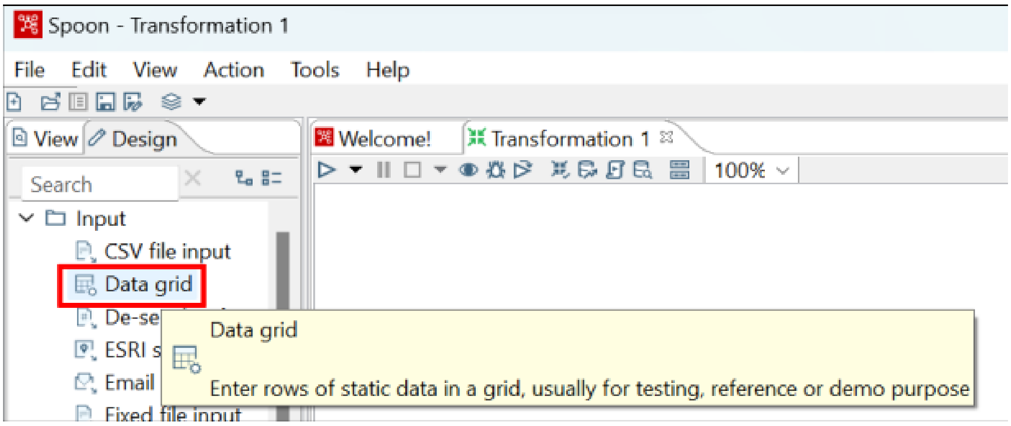
Examine preview data

Rows of step: FULL OUTER (12 rows)

| # | EmpID | EMPName | DepNo | DepNo_1 | DepName |
|---|---|---|---|---|---|
| 1 | 112.0 | Makya | 5.0 | <null> | <null> |
| 2 | 111.0 | Adi | 4.0 | <null> | <null> |
| 3 | 109.0 | Tonny | 3.0 | 3.0 | PURCHASE |
| 4 | 106.0 | Rock | 3.0 | 3.0 | PURCHASE |
| 5 | 103.0 | Rocky | 3.0 | 3.0 | PURCHASE |
| 6 | 110.0 | Yashu | 2.0 | 2.0 | MARKETING |
| 7 | 107.0 | Raj | 2.0 | 2.0 | MARKETING |
| 8 | 104.0 | Ronnie | 2.0 | 2.0 | MARKETING |
| 9 | 102.0 | Tom | 2.0 | 2.0 | MARKETING |
| 10 | 108.0 | Sid | 1.0 | 1.0 | IT |
| 11 | 105.0 | Roy | 1.0 | 1.0 | IT |
| 12 | 101.0 | John | 1.0 | 1.0 | IT |

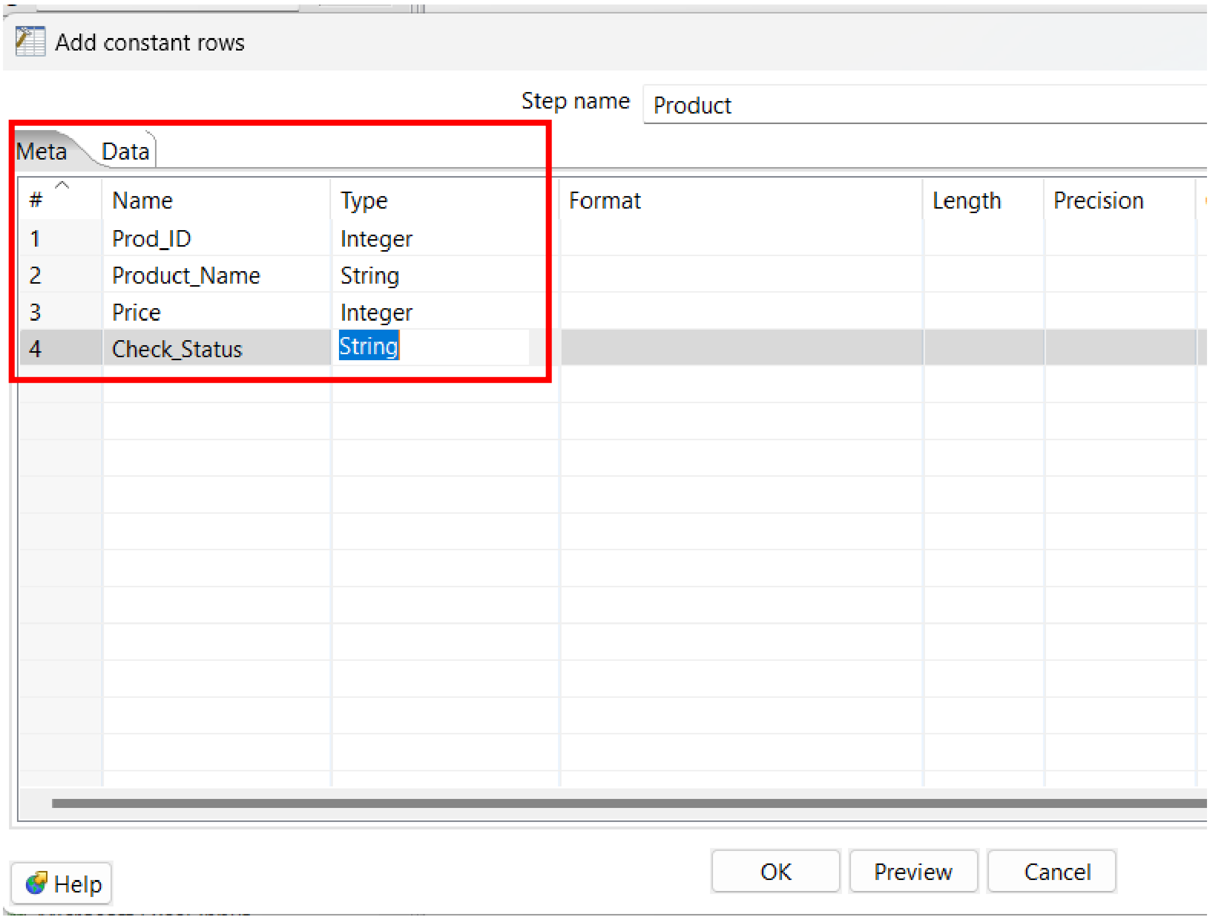
## TRANSFORMATION 9: Data validations

**Step 1: -** Drag and drop Data Grid from Input folder under Design tab.



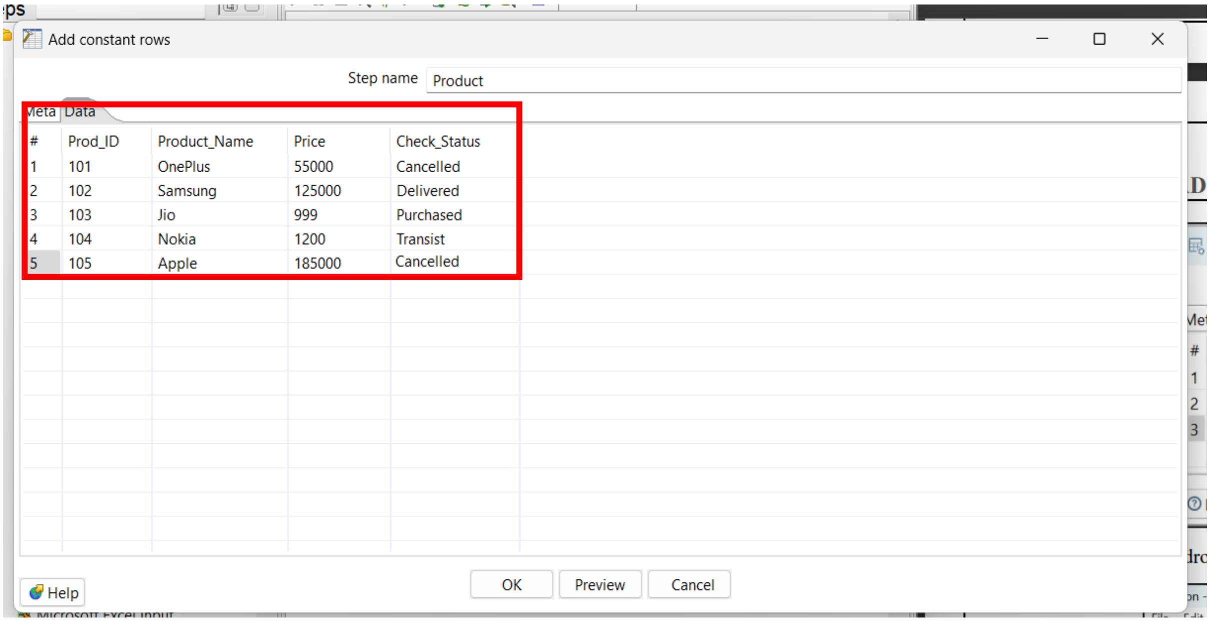
Rename it as Product.

**Step 2: -** Double click on Product data grid and insert records as shown below→ Click on OK.



Step name: Product

| # | Name | Type | Format | Length | Precision |
|---|---|---|---|---|---|
| 1 | Prod_ID | Integer | | | |
| 2 | Product_Name | String | | | |
| 3 | Price | Integer | | | |
| 4 | Check_Status | String | | | |



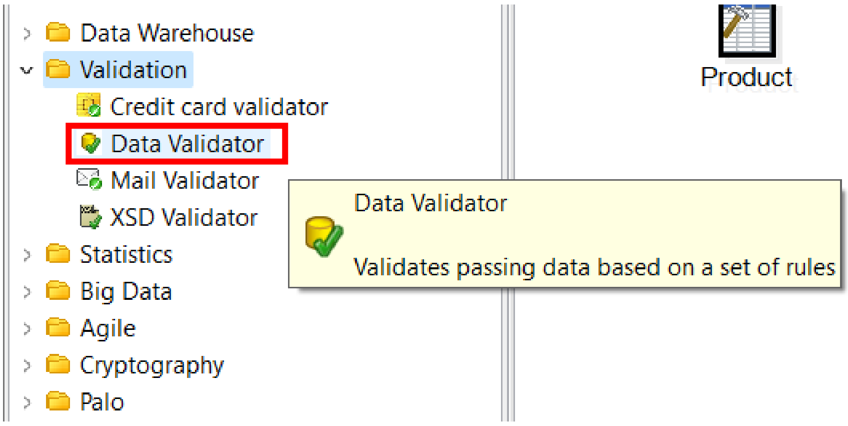
Add constant rows

Step name: Product

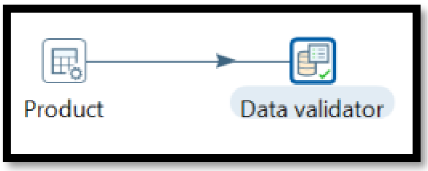
Meta | Data

| # | Prod_ID | Product_Name | Price | Check_Status |
|---|---|---|---|---|
| 1 | 101 | OnePlus | 55000 | Cancelled |
| 2 | 102 | Samsung | 125000 | Delivered |
| 3 | 103 | Jio | 999 | Purchased |
| 4 | 104 | Nokia | 1200 | Transist |
| 5 | 105 | Apple | 185000 | Cancelled |

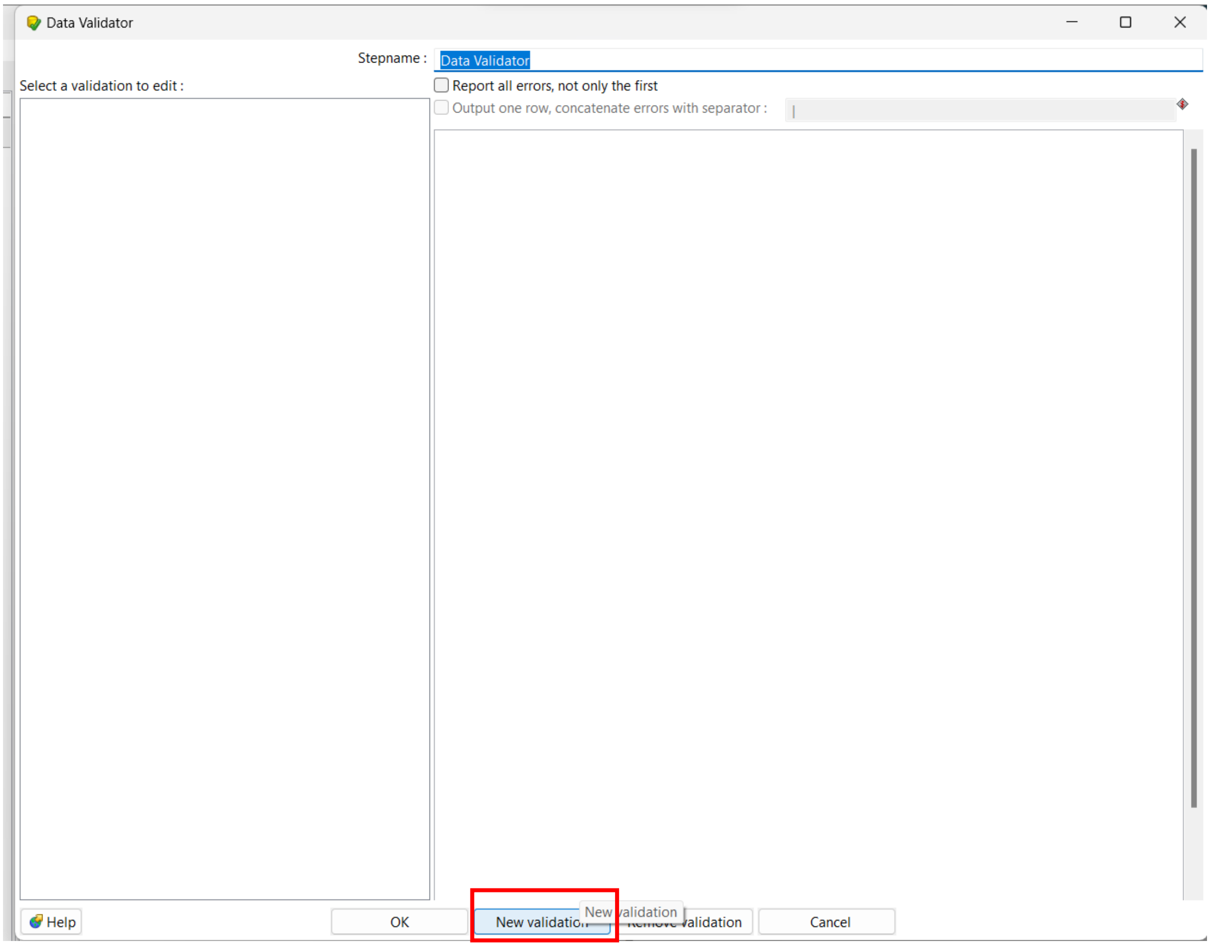
**Step 3: -** Drag and drop Data validator from Validation folder under Design tab.



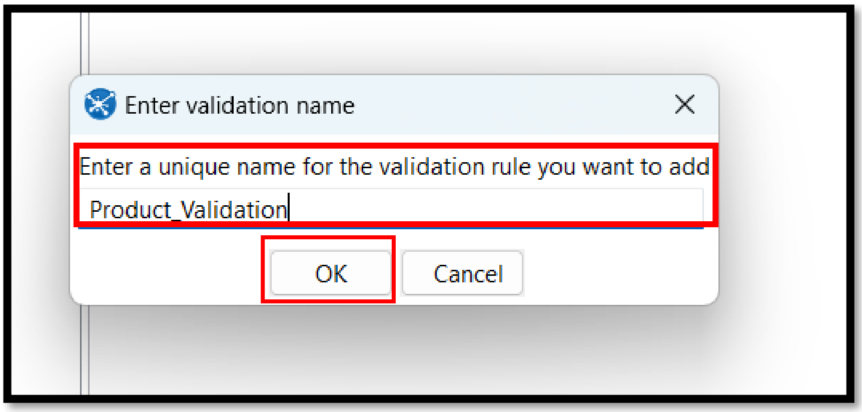
Hold the mouse pointer on Product data grid and select and drag the output connector to the Data validator



Double clickc on Data validator→New Validattion


Data Validator
Stepname : Data Validator
Select a validation to edit :
☐ Report all errors, not only the first
☐ Output one row, concatenate errors with separator :
Help
OK
New validation
New validation
Remove validation
Cancel

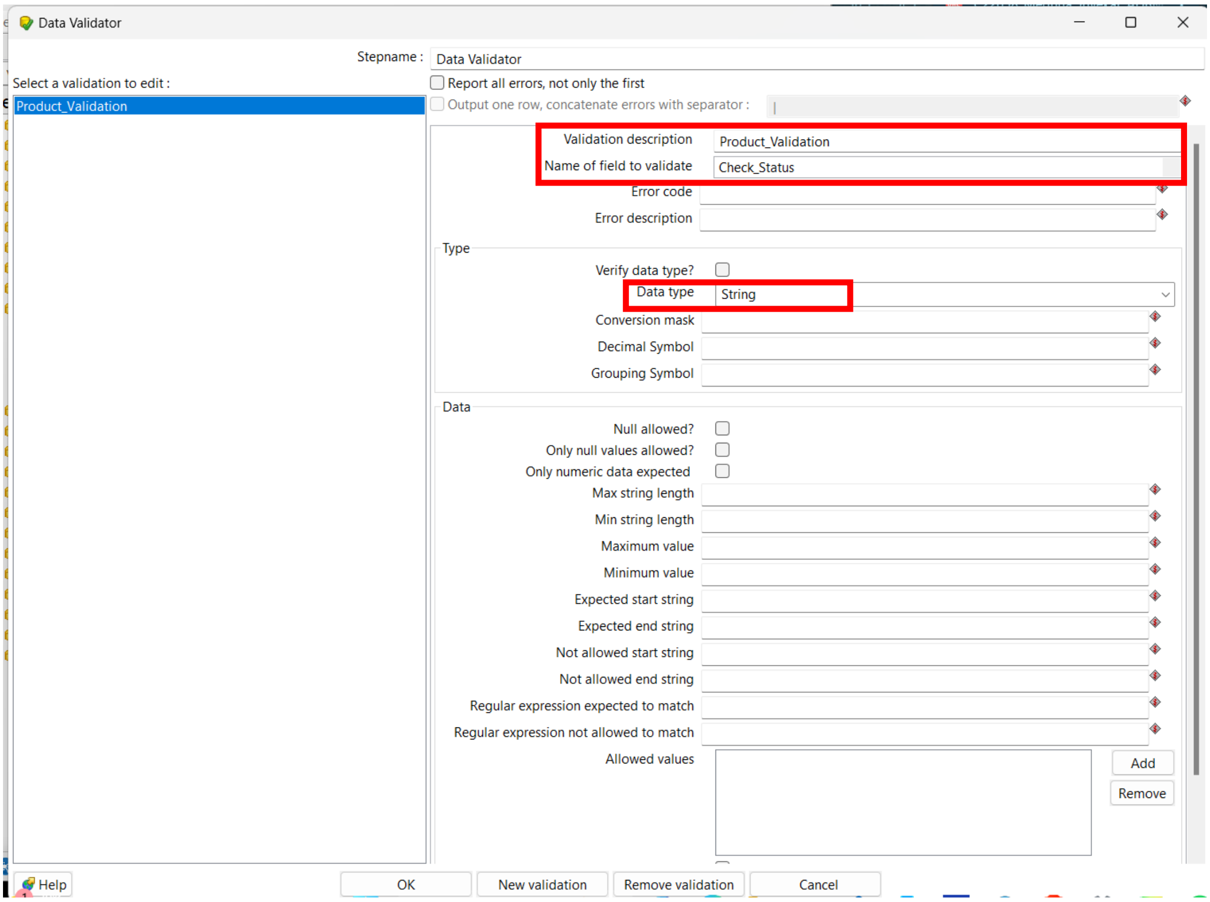
Give Valiadtion Name and click OK.



Enter a unique name for the validation rule you want to add

Product_Validation

Select the validation to edit and fill in the details as shown below.



Data Validator

Stepname : Data Validator

Select a validation to edit : Product_Validation

□ Report all errors, not only the first

□ Output one row, concatenate errors with separator :

Validation description: Product_Validation

Name of field to validate: Check_Status

Error code:

Error description:

Type

Verify data type? □

Data type: String

Conversion mask:

Decimal Symbol:

Grouping Symbol:

Data

Null allowed? □

Only null values allowed? □

Only numeric data expected □

Max string length:

Min string length:

Maximum value:

Minimum value:

Expected start string:

Expected end string:

Not allowed start string:

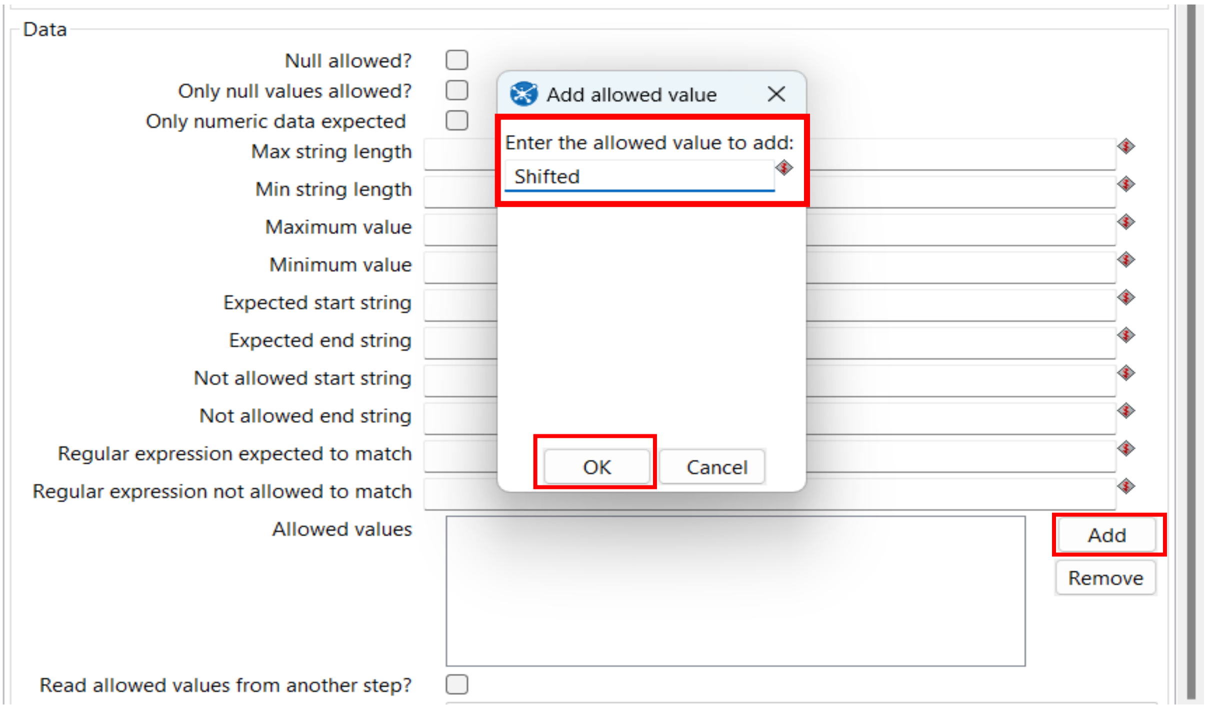
Not allowed end string:

Regular expression expected to match:

Regular expression not allowed to match:

Allowed values:

Click on add to set validation, set it as Shifted & press Enter and click on ok



Data

Null allowed? □

Only null values allowed? □

Only numeric data expected □

Max string length

Min string length

Maximum value

Minimum value

Expected start string

Expected end string

Not allowed start string

Not allowed end string

Regular expression expected to match

Regular expression not allowed to match

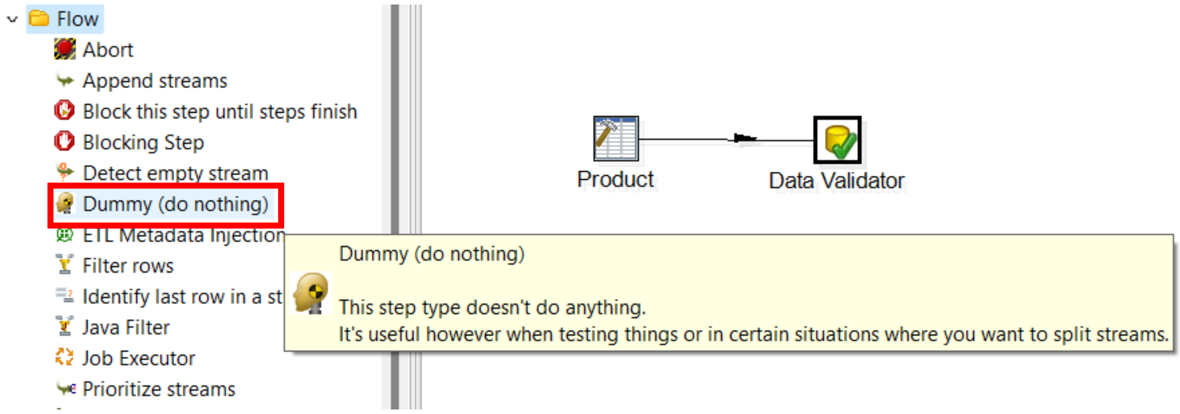
Allowed values

Read allowed values from another step? □

Add allowed value

Enter the allowed value to add: Shifted

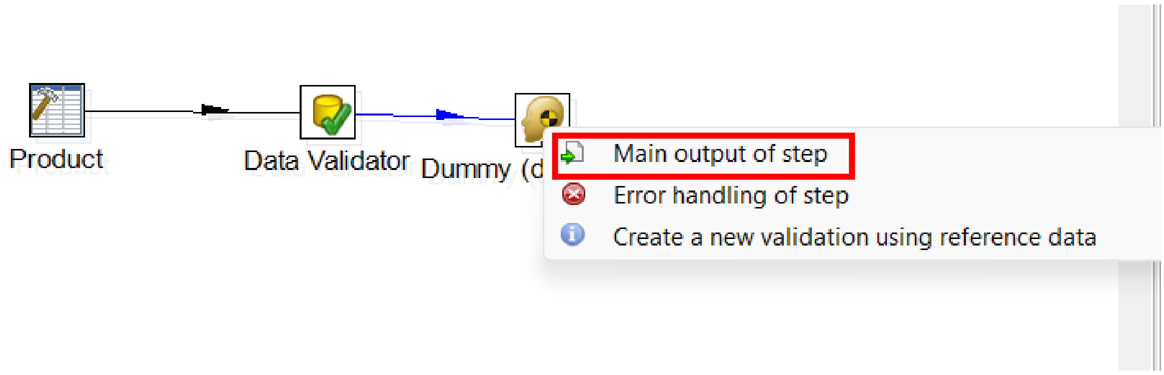
Click on OK. Step 4: Drag and drop Dummy from Flow folder under the Design tab.



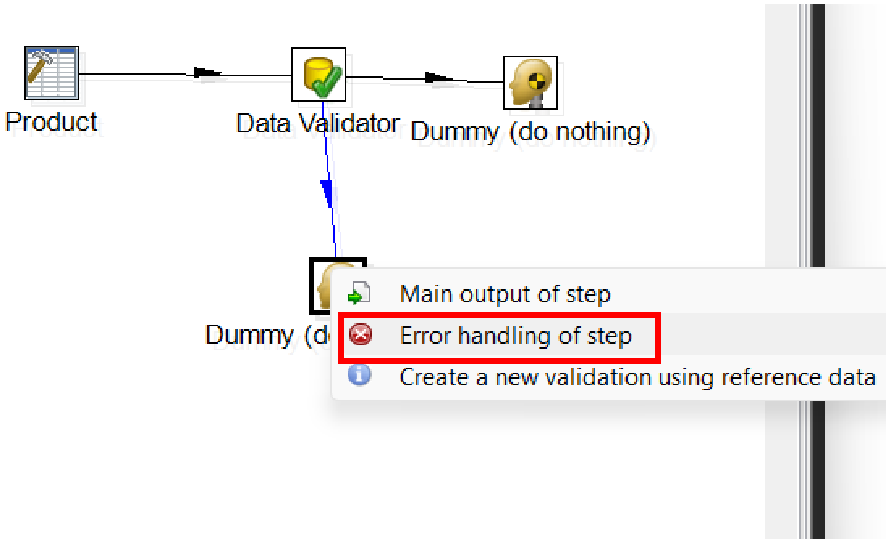
Dummy (do nothing)

This step type doesn't do anything.
It's useful however when testing things or in certain situations where you want to split streams.

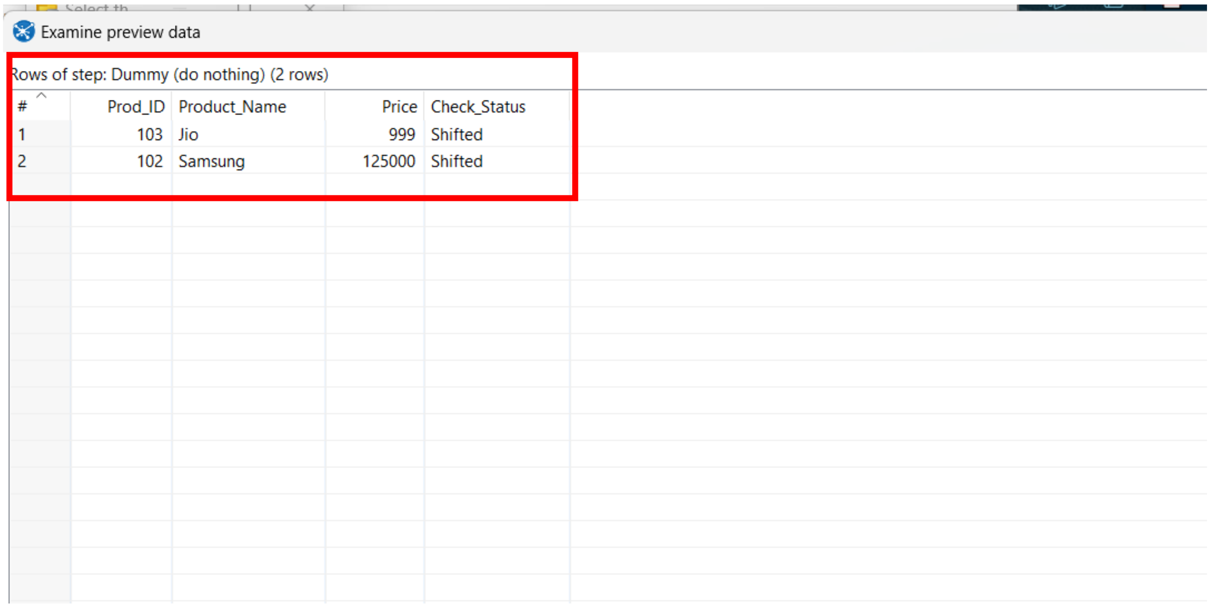
Hold the mouse pointer on Data validator and select and drag the output connector to the Dummy. Select Main output of step.



**Step 5: -** Drag and drop another Dummy from Flow folder under the Design tab and connect it to the data validator. Select Error handling of step. In the next window click in Copy.



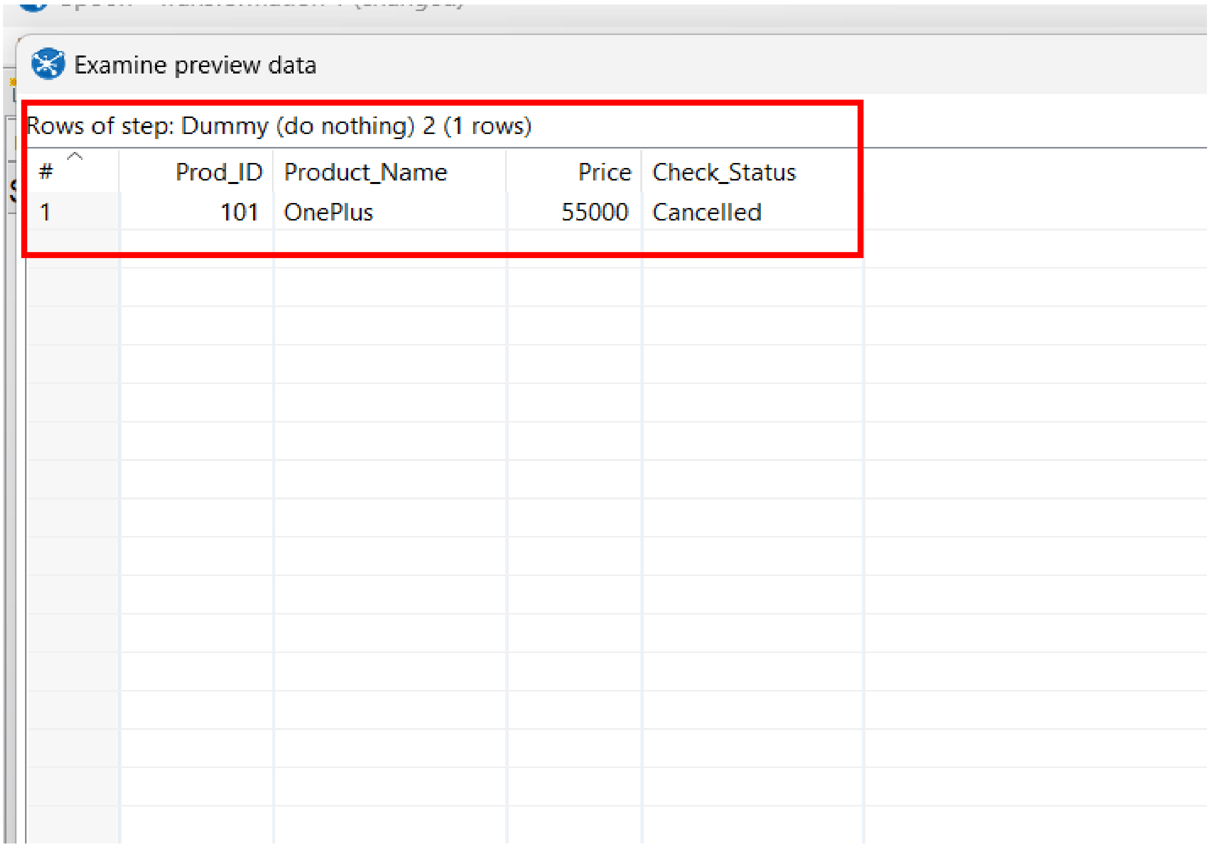
**Step 6: -** Quick Launch the transformation selecting one dummy file each.



Examine preview data

Rows of step: Dummy (do nothing) (2 rows)

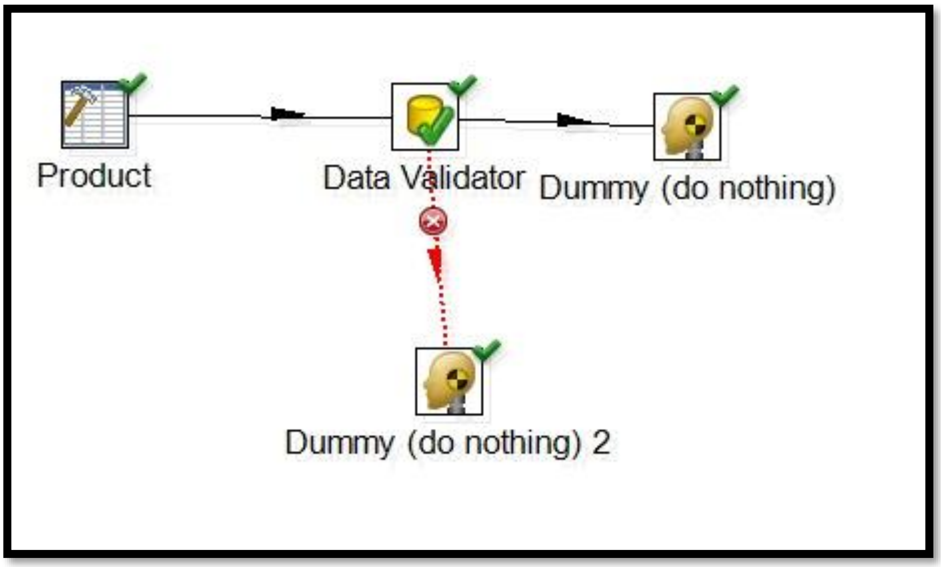
| # | Prod_ID | Product_Name | Price | Check_Status |
|---|---|---|---|---|
| 1 | 103 | Jio | 999 | Shifted |
| 2 | 102 | Samsung | 125000 | Shifted |



Examine preview data

Rows of step: Dummy (do nothing) 2 (1 rows)

| # | Prod_ID | Product_Name | Price | Check_Status |
|---|---|---|---|---|
| 1 | 101 | OnePlus | 55000 | Cancelled |


Product
Data Validator
Dummy (do nothing)
Dummy (do nothing) 2

# Practical No.: 5

# Introduction To R Programming

```
> x = 1
> print(x)
[1] 1
> class(x)
[1] "numeric"
> y="c"
> print(y)
[1] "c"
> is.character(y)
[1] TRUE
```

## Vector Operations

```
> x = c(11.8,24.7,37.9)
> y=c(8,7,3)
> x*y
[1]  94.4 172.9 113.7
> x-y
[1]  3.8 17.7 34.9
> x+y
[1] 19.8 31.7 40.9
># using matrix() function
> m = matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3)
> m
     [,1] [,2] [,3]
[1,]   11   55   66
[2,]   12   60   72
[3,]   13   65   78
> dim(m)
```

```
[1] 3 3
> m = matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3,byrow = TRUE)
> m
     [,1] [,2] [,3]
[1,]   11   12   13
[2,]   55   60   65
[3,]   66   72   78
```

## cbind column bind and rbind row bind

```
>x=c(3,2,9)
> y=c(12,13,14)
> cbind(x,y)
     x  y
[1,] 3 12
[2,] 2 13
[3,] 9 14
>rbind(x,y)
  [,1] [,2] [,3]
x    3    2    9
y   12   13   14
```

## Matrix Multiplication

```
> p = 3*m
> p
     [,1] [,2] [,3]
[1,]   33   36   39
[2,]  165  180  195
[3,]  198  216  234
> n = matrix(c(4,5,6,13,14,15,24,25,26),nrow = 3,ncol = 3)
> q = m+n
> q
```

```
     [,1] [,2] [,3]
[1,]   15   25   37
[2,]   60   74   90
[3,]   72   87  104
> mdash=t(m)
> mdash
     [,1] [,2] [,3]
[1,]   11   55   66
[2,]   12   60   72
[3,]   13   65   78
```

## Determinant

```
>s = matrix(c(2,3,4,14,15,16,21,22,23),nrow=3,ncol=3,byrow = TRUE)
> s_det = det(s)
> s_det
[1] 2.109424e-15
> r_det = det(m)
> r_det
[1] 0
```

## DataFrame

```
>student_id = c(1,2,3)
> student_names = c("Krutarth","Chandresh","Aditya")
> position = c("First","Second","Third")
> data = data.frame(student_id,student_names,position)
> data
  student_id  student_names   position
1          1       Krutarth      First
2          2      Chandresh     Second
3          3         Aditya      Third
```

```
> data$student_id [1]
[1] 1
> names(data)
[1] "student_id"    "student_names" "position"
```

## Table Command

```
> smoke = matrix(c(51,43,22,92,28,21,68,22,9),ncol=3,byrow=TRUE)
> colnames(smoke) = c("High","Low","Middle")
> rownames(smoke) = c("current","former","never")
> smoke = as.table(smoke)
> smoke
            High   Low    Middle
current      51     43      22
former      92     28      21
never        68     22      9
```

## Operation in csv file

```
>cs = read.table("ccV1.csv",sep = ",", header = T)
>cs
Emp_no   Emp_name   Salary
1 101      Krutarth     12000
2 102      Aditya       12400
3 103      Chandresh      10000
>        # dimension
>        dim(cs) [1] 3 3
>        head(cs,2)
Emp_no   Emp_name   Salary
1 101      Krutarth       12000
2 107       Aditya        9400
```

# Load first two lines at the bottom

```
>       tail(cs,2)

        Emp_no   Emp_name    Salary
 2       107      Aditya      9400
4       108     Krutarth     5850
```

## Operation in xlsx file

```
>       csvv = XLConnect::readWorksheetFromFile("demo.xlsx",sheet=1)
>       csvv
RollNo   Name      Marks
21         Krutarth    88
22         Aditya      79
23      Chandresh    84
```

## Practical No.: 06

## Implementation of Data PreProcessing Techniques :

- **Naming & Renaming Varaiables , Adding a new variables**
- **Dealing with missing value**
- **Dealing with categorical data**
- **Data reduction using subsetting**

```
> library(dplyr)

Attaching package: ‘dplyr’

The following objects are masked from ‘package:stats’:

    filter, lag

The following objects are masked from ‘package:base’:

    intersect, setdiff, setequal, union

> data <- data.frame(
+ ID = 1:5,
+    Name = c("Krutarth", "Aditya", "Chandresh", "Noel", "Atharv"),
+ Age = c(21, 23, 20, NA, 28),
+ Score = c(85, 92, 78, 65, 89))
>  print(data)
  ID     Name       Age    Score
1  1     Krutarth    21      85
2  2      Aditya     23      92
3  3    Chandresh    20      78
4  4     Noel        NA      65
5  5    Atharv        28      89
>  data <- data %>%
+    rename(UserID = ID, FullName = Name)
>  print(data)
```

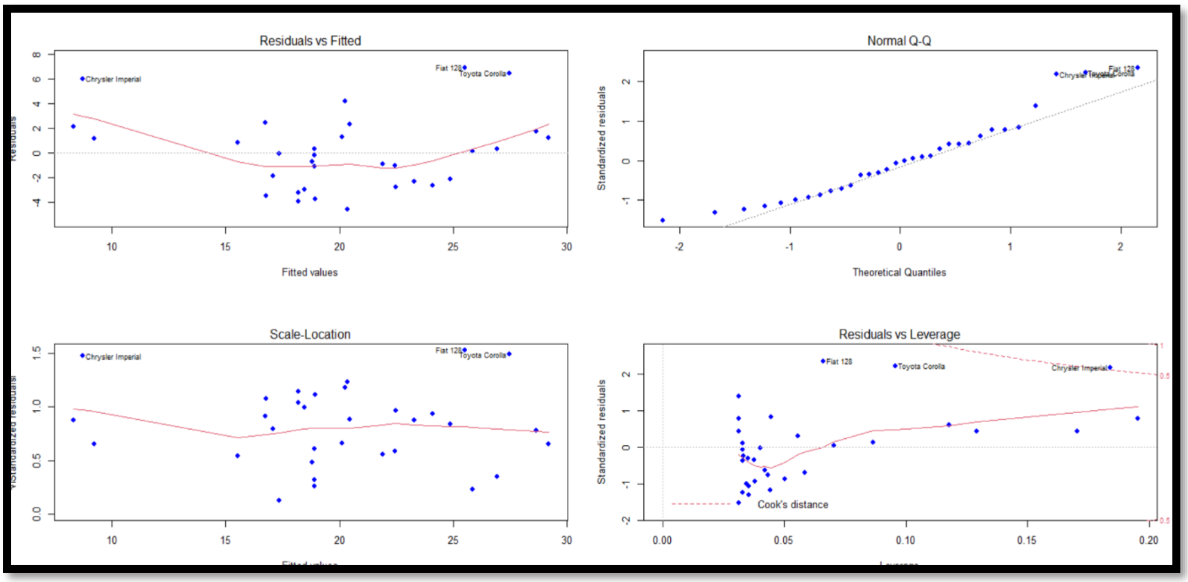
```
  UserID  FullName   Age  Score
1    1     Krutarth    21    85
2    2     Aditya      23    92
3    3    Chandresh    20    78
4    4    Noel         NA    65
5    5    Atharv       28    89
> data$Grade <- ifelse(data$Score >= 90, "A", ifelse(data$Score >= 80, "B", "C"))
>  print(data)
  UserID  FullName   Age  Score  Grade
1    1     Krutarth   21     85      B
2    2      Aditya    23     92      A
3    3   Chandresh    20     78      C
4    4    Noel        NA     65      C
5    5   Atharv        28     89      B
> data <- na.omit(data)
>  print(data)
  UserID  FullName   Age  Score  Grade
1    1     Krutarth    21    85      B
2    2      Aditya     23    92      A
3    3   Chandresh     20    78      C
5    5     Atharv      28    89      B
> data$Gender <- factor(c("Male", "Male", "Male", "Male"))
>  print(data)
  UserID  FullName   Age  Score  Grade  Gender
1    1     Krutarth    21    85      B     Male
2    2     Aditya      23    92      A     Male
3    3   Chandresh    20     78       C    Male
5    5     Atharv      28     89      B     Male
> subset_data <- subset(data, Age > 20)
>  print(data)
```

```
  UserID  FullName  Age  Score  Grade  Gender
1    1     Krutarth   21    85     B     Male
2    2     Aditya     23    92     A     Male
3    3    Chandresh   20    78     C     Male
5    5     Atharv     28    89     B     Male
> print(subset_data)

  UserID  FullName  Age  Score  Grade  Gender
1    1    Krutarth   21    85     B    Male
2    2     Aditya    23    92     A    Male
5    5     Atharv    28    89     B    Male
```

## Practical .: 7

## Implementation & Analysis of Linear Regression through graphical methods.

```
> data(mtcars)
> model <- lm(mpg ~ wt, data = mtcars)
> summary(model)
Call:
lm(formula = mpg ~ wt, data = mtcars)
Residuals:
    Min      1Q  Median      3Q     Max
-4.5432 -2.3647 -0.1252  1.4096  6.8727
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
wt           -5.3445     0.5591  -9.559 1.29e-10 ***
---
Signif. codes:  0 ‘’ 0.001 ‘’ 0.01 ‘’ 0.05 ‘.’ 0.1 ‘ ’ 1
Residual standard error: 3.046 on 30 degrees of freedom
Multiple R-squared:  0.7528,    Adjusted R-squared:  0.7446
F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
> plot(mtcars$wt, mtcars$mpg, main = "Scatterplot of Weight vs. MPG with Regression Line",
+      xlab = "Weight", ylab = "Miles Per Gallon", pch = 16, col = "blue")
> abline(model, col = "red", lwd = 2)
> par(mfrow = c(2, 2))
> plot(model, pch = 16, col = "blue")
> par(mfrow = c(1, 1))
```

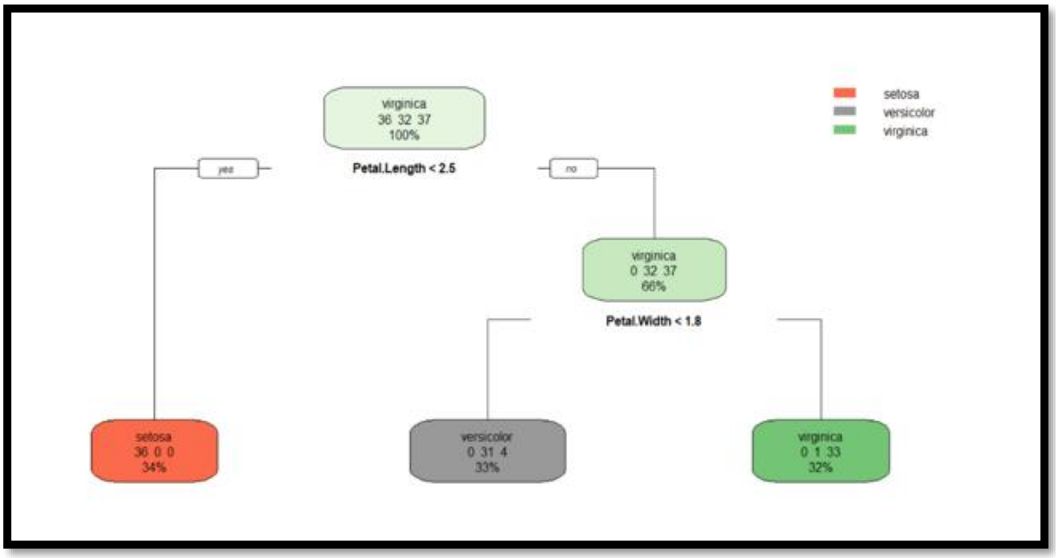

Residuals vs Fitted
Residuals
Chrysler Imperial
Fiat 128
Toyota Corolla
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Scale-Location
√Standardized residuals
Residuals vs Leverage
Cook's distance
1
0.5
Leverage

# Practical No.: 8

## Implementation & Analysis Classification algorithms like Naïve Bayesian, K-Nearest Neighbour, ID3, C4.5.

### Decision Tree :

```
> library(class)
>  library(e1071)
>  library(rpart)
> library(C50)
> library(ggplot2)
> library(ROCR)
> data(iris)
> set.seed(123)
>  train_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
> train_data <- iris[train_indices, ]
>  test_data <- iris[-train_indices, ]
> dt_model <- rpart(Species ~ ., data = train_data, method = "class")
> dt_pred <- predict(dt_model, test_data, type = "class")
> dt_accuracy <- sum(dt_pred == test_data$Species) / nrow(test_data)
> cat("Decision Tree Accuracy:", dt_accuracy, "\n")
Decision Tree Accuracy: 0.9777778
> print(summary)
function (object, ...)
UseMethod("summary")
<bytecode: 0x0000000016fb2d20>
<environment: namespace:base>
> rpart.plot(dt_model, type = 2, extra = 101)
```



**Decision Tree Plot**

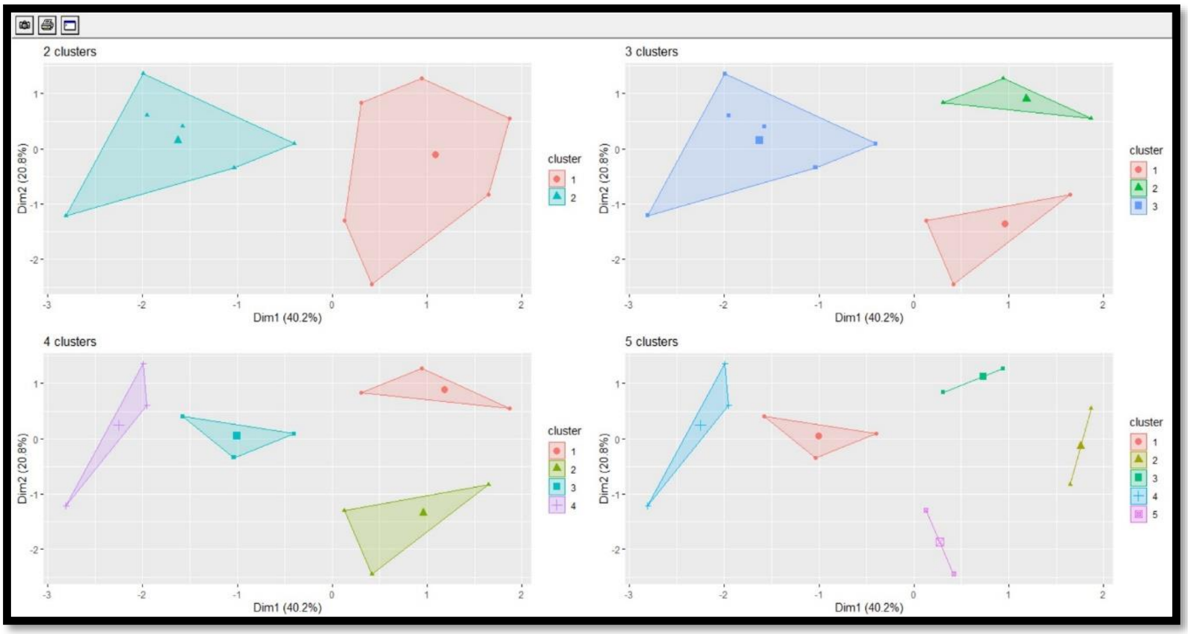
## Nayive Bayesian :

```
> library(class)
> library(e1071)
> library(rpart)
> library(rpart.plot)
> library(C50)
> library(ggplot2)
> library(ROCR)
> data(iris)
> set.seed(123)
> nb_model <- naiveBayes(Species ~ ., data = train_data)
> train_indices <- sample(1:nrow(iris), 0.7 * nrow(iris))
>  train_data <- iris[train_indices, ]
> test_data <- iris[-train_indices, ]
> nb_model <- naiveBayes(Species ~ ., data = train_data)
> nb_pred <- predict(nb_model, test_data[, -5])
> nb_accuracy <- sum(nb_pred == test_data$Species) / nrow(test_data)
>  cat("Naive Bayes Accuracy:", nb_accuracy, "\n")
```

```
Naive Bayes Accuracy: 1
> summary(nb_model)
          Length Class  Mode
apriori   3      table  numeric
tables    4      -none- list
levels    3      -none- character
isnumeric 4      -none- logical
call      4      -none- call
> nb_table <- table(Actual = test_data$Species, Predicted = nb_pred)
> ggplot(as.data.frame.table(nb_table), aes(x = Actual, y = Predicted, fill = Freq)) +
+ geom_tile() +
+ labs(title = "Naive Bayes Confusion Matrix",
+        x = "Actual",
+ y = "Predicted")
```

## K-Nearest :

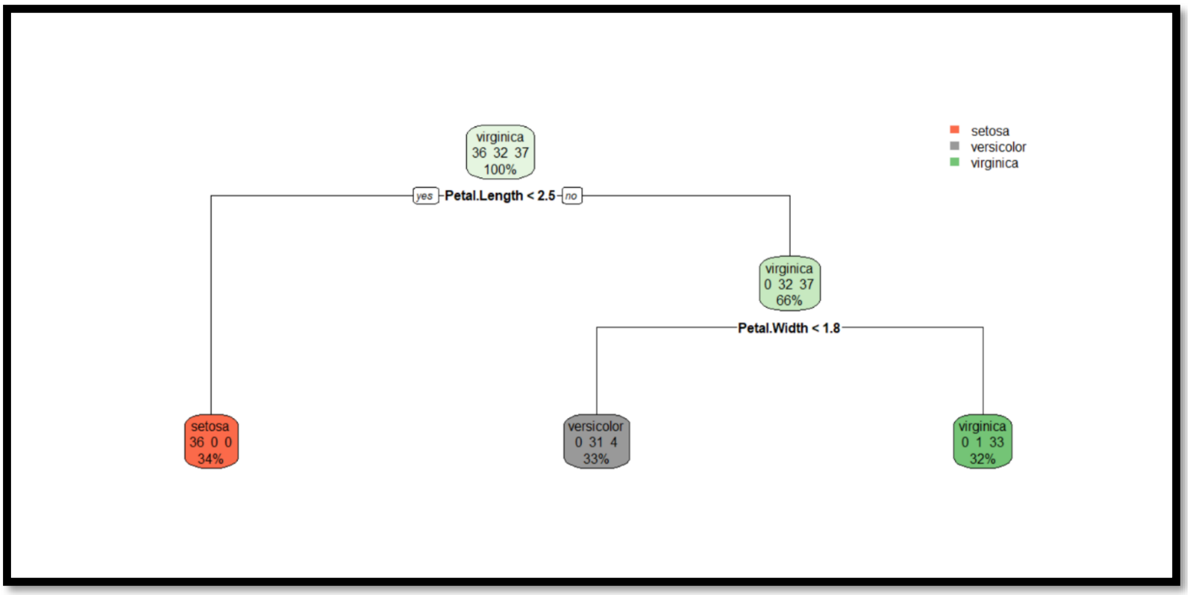
```
install.packages("cluster")
install.packages("factoextra")
install.packages("gridextra")
library(cluster)
> library(factoextra)
> data<-animals
> data<-na.omit(data)
> head(data,n=10)
      war  fly ver  end  gro  hai
ant    1    1    1   1    2    1
bee    1    2    1   1    2    2
cat    2    1    2   1    1    2
cpl    1    1    1   1    1    2
chi    2    1    2   2    2    2
cow    2    1    2   1    2    2
duc    2    2    2   1    2    1
eag    2    2    2   2    1    1
ele    2    1    2   2    2    1
fly    1    2    1   1    1    1
> kn<-kmeans(data,centers=2,nstart=25)
> kn2<-kmeans(data,centers=3,nstart=25)
> kn3<-kmeans(data,centers=4,nstart=25)
> kn4<-kmeans(data,centers=5,nstart=25)
> plot1<-fviz_cluster(kn,geom="point",data=data)+ggtitle("2 clusters")
> plot2<-fviz_cluster(kn2,geom="point",data=data)+ggtitle("3 clusters")
> plot3<-fviz_cluster(kn3,geom="point",data=data)+ggtitle("4 clusters")
> plot4<-fviz_cluster(kn4,geom="point",data=data)+ggtitle("5 clusters")
> library(gridExtra)
> grid.arrange(plot1,plot2,plot3,plot4,nrow=2)
```



# K-Nearest Plot

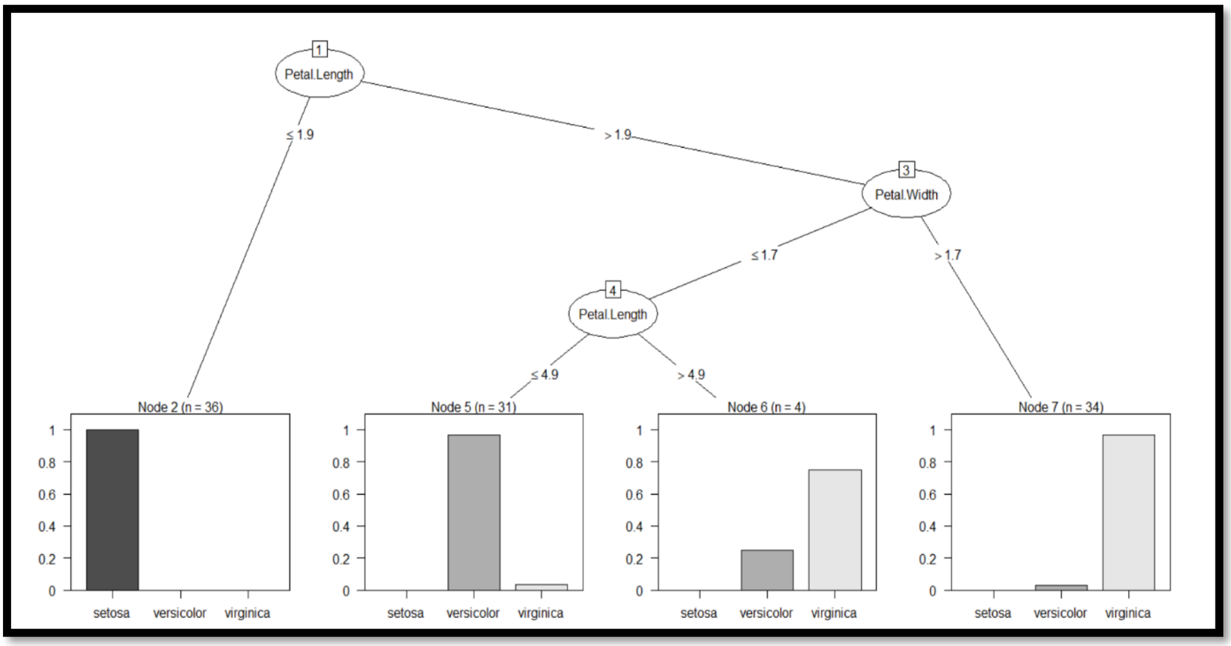
## ID 3 :

```
> id3_model <- rpart(Species ~ ., data = train_data, method = "class")
>  id3_pred <- predict(id3_model, test_data, type = "class")
>  id3_accuracy <- sum(id3_pred == test_data$Species) / nrow(test_data)
> cat("ID3 Accuracy:", id3_accuracy, "\n")
ID3 Accuracy: 0.9777778
> rpart.plot(id3_model, type = 2, extra = 101)
```



## C4.5 :

```
> id3_model <- rpart(Species ~ ., data = train_data, method = "class")
>  id3_pred <- predict(id3_model, test_data, type = "class")
>  id3_accuracy <- sum(id3_pred == test_data$Species) / nrow(test_data)
> cat("ID3 Accuracy:", id3_accuracy, "\n")
ID3 Accuracy: 0.9777778
> rpart.plot(id3_model, type = 2, extra = 101)
> c45_model <- C5.0(train_data[, -5], train_data$Species)
>  c45_pred <- predict(c45_model, newdata = test_data[, -5])
>  c45_accuracy <- sum(c45_pred == test_data$Species) / nrow(test_data)
> cat("C4.5 Accuracy:", c45_accuracy, "\n")
C4.5 Accuracy: 0.9777778
>  plot(c45_model)
```



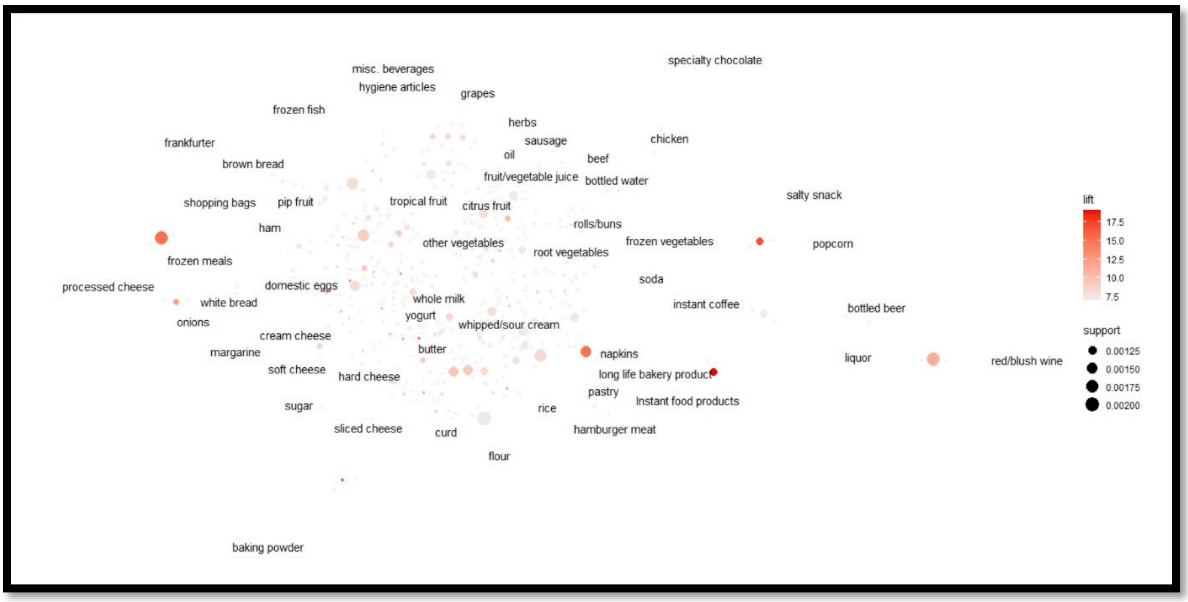
**C 4.5 Plot**

# Practical .: 9

## Implementation & Analysis of Apriori Algorithm using Market Basket Analysis

```
> library(arules)
Loading required package: Matrix
Attaching package: ‘arules’
The following objects are masked from ‘package:base’:
    abbreviate, write
>  library(arulesViz)
> data("Groceries")
>  summary(Groceries)
transactions as itemMatrix in sparse format with
 9835 rows (elements/itemsets/transactions) and
 169 columns (items) and a density of 0.02609146
most frequent items:
      whole milk other vegetables       rolls/buns             soda
            2513             1903             1809             1715
          yogurt          (Other)
            1372            34055
element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46
  17   18   19   20   21   22   23   24   26   27   28   29   32
  29   14   14    9   11    4    6    1    1    1    1    3    1
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   4.409   6.000  32.000
includes extended item information - examples:
       labels  level2           level1
1 frankfurter sausage meat and sausage
2     sausage sausage meat and sausage
3  liver loaf sausage meat and sausage
> rules <- apriori(Groceries, parameter = list(support = 0.001, confidence = 0.5))
Apriori
Parameter specification:
 confidence minval smax arem  aval originalSupport maxtime support minlen
        0.5    0.1    1 none FALSE            TRUE       5   0.001      1
 maxlen target  ext
     10  rules TRUE
Algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE
Absolute minimum support count: 9
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[169 item(s), 9835 transaction(s)] done [0.00s].
sorting and recoding items ... [157 item(s)] done [0.00s].
creating transaction tree ... done [0.02s].
checking subsets of size 1 2 3 4 5 6 done [0.01s].
writing ... [5668 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].
> plot(rules, method = "graph", control = list(type = "items"))
Warning: Unknown control parameters: type
Available control parameters (with default values):
layout   =  stress
circular        =  FALSE
ggraphdots       =  NULL
edges    =  <environment>
```

```
nodes    = <environment>
nodetext        = <environment>
colors   = c("#EE0000FF", "#EEEEEEFF")
engine   =  ggplot2
max      =  100
verbose  =  FALSE
Warning message:
Too many rules supplied. Only plotting the best 100 using ‘lift’ (change control parameter max if needed).
Error in UseMethod("depth") :
  no applicable method for 'depth' applied to an object of class "NULL"
Error in diff.default(xscale) :
  VECTOR_ELT() can only be applied to a 'list', not a 'raw'
```



## Practical .:10

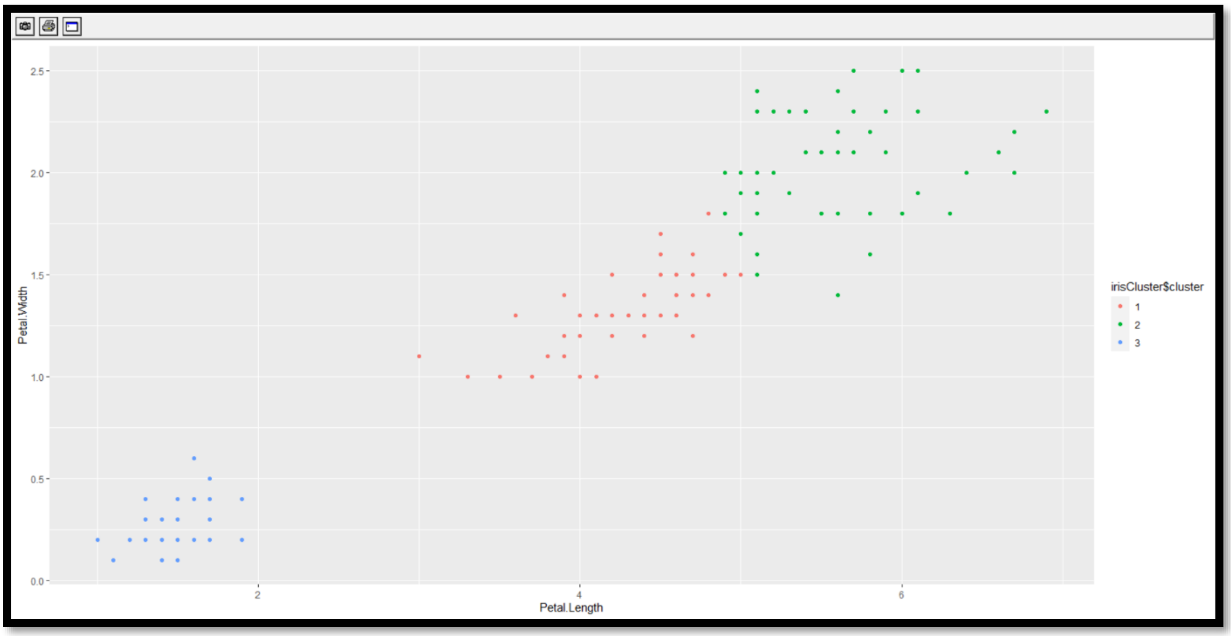
## Implementation & analysis of clustering algorithms like K-means & Agglomarative , Divisive.

```
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1         5.1         3.5          1.4         0.2  setosa
2         4.9         3.0          1.4         0.2  setosa
3         4.7         3.2          1.3         0.2  setosa
4         4.6         3.1          1.5         0.2  setosa
5         5.0         3.6          1.4         0.2  setosa
6         5.4         3.9          1.7         0.4  setosa
> ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom_point()
Error in ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) :
  could not find function "ggplot"
> library(ggplot2)
> ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom_point()
> set.seed(20)
>  irisCluster <- kmeans(iris[, 3:4], 3, nstart = 20)
>  irisCluster
K-means clustering with 3 clusters of sizes 52, 48, 50
```

**Cluster means:**

```
  Petal.Length Petal.Width
1     4.269231    1.342308
2     5.595833    2.037500
3     1.462000    0.246000
Clustering vector:
  [1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
 [44] 3 3 3 3 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 2 1 1
 [87] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 1 2 2
[130] 2 2 2 2 2 2 2 2 2 1 2 2 2 2 2 2 2 2 2 2 2
```

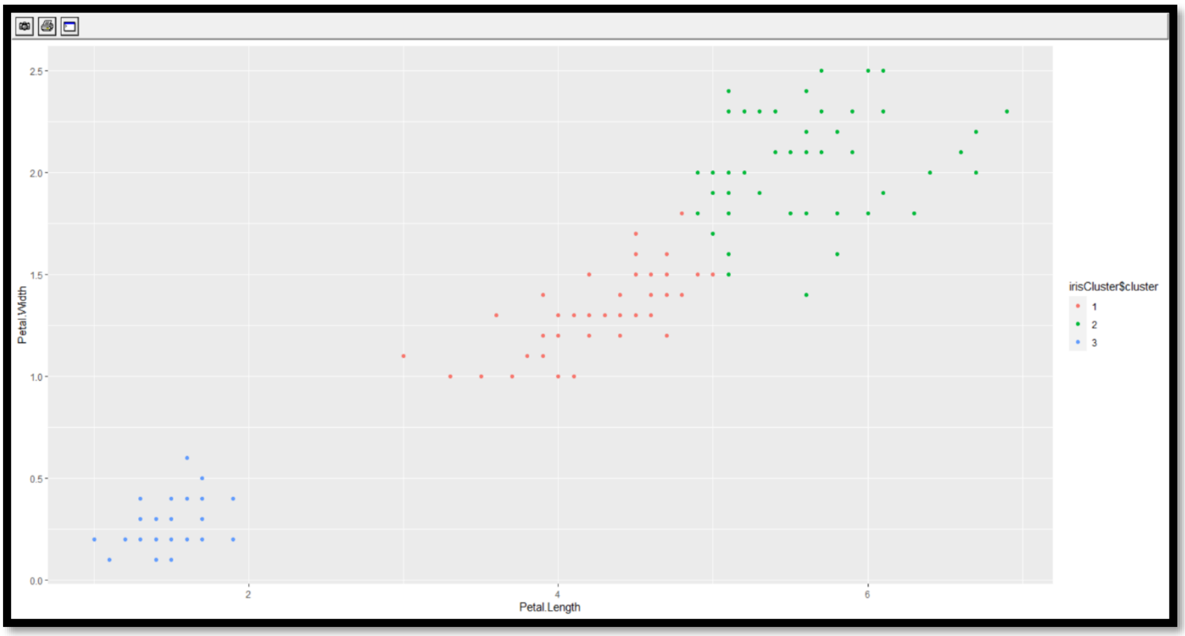
```
Within cluster sum of squares by cluster:
[1] 13.05769 16.29167  2.02200
 (between_SS / total_SS =  94.3 %)
Available components:
[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
[6] "betweenss"    "size"         "iter"         "ifault"
> irisCluster$cluster <- as.factor(irisCluster$cluster)
> ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) + geom_point()
```



## Agglomerative :

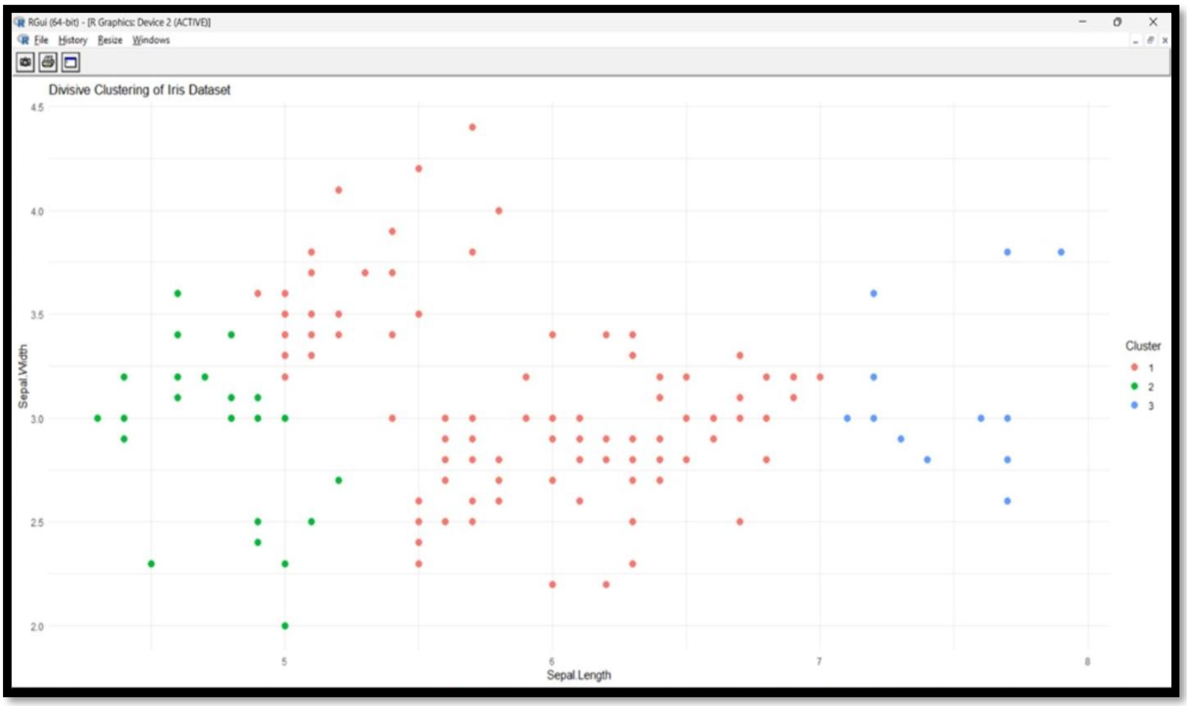
```
> clusters <- hclust(dist(iris[,3:4]), method = 'average')
>  plot(clusters)
> clusterCut <- cutree(clusters, 3)
> table(clusterCut, iris$Species)
```

```
clusterCut   setosa   versicolor   virginica
        1        50          0           0
        2         0         45           1
        3         0          5          49
```



## Divisive :

```
> eatures <- iris[, c("Sepal.Length", "Sepal.Width")]

> hclust_result <- hclust(dist(features), method = "complete")

> num_clusters <- 3

> clusters <- cutree(hclust_result, k = num_clusters)

> iris$cluster <- as.factor(clusters)

> ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = cluster))+geom_point(size = 3)+
ggtitle("Divisive Clustering of Iris Dataset")+labs(color = "Cluster")+ theme_minimal()
```


RGui (64-bit) - [R Graphics: Device 2 (ACTIVE)]
File History Resize Windows
Divisive Clustering of Iris Dataset
4.5
4.0
3.5
3.0
2.5
2.0
Sepal.Width
5
6
7
8
Sepal.Length
Cluster
1
2
3