Deccan Education Society's

Navinchandra Mehta Institute of Technology and Development

CERTIFICATE

This is to certify that Mr. **Mehul Parihar** of M.C.A.

Semester I with Roll No.**C22091** has completed. Practicals of

ADVANCED DATABASE SYSTEMS under my supervision in this

college during the year 2022 -2023.

CO	R1 (Attendance)	R2 (Performance during lab session)	R3 (Innovation in problem-solving technique)	R4 (Mock Viva)	R5 (Variation in implementation of learned topics on projects)
CO1					
CO2					
CO3					
CO4					

Practical-in-charge

Head of Department
MCA Department
(NMITD)

Index

	Attaching and Detaching data. Reading data from the consol. Loading data from different data sources.(CSV, Excel).		
6	Implementation of Data preprocessing techniques like, Naming and Renaming variables, adding a new variable. Dealing with missing data. Dealing with categorical data. Data reduction using subsetting	02-03-2022	
7	Implementation and analysis of Classification algorithms like Naive Bayesian, K-Nearest Neighbor, ID3 , C4.5	11-03-2022	
8	Implementation and analysis of Apriori Algorithm using Market Basket Analysis.	16-03-2022	
9	Implementation and analysis of clustering algorithms like K-Means , Agglomerative	19-03-2022	

1. Implementation of different types of Partitions : Range, List and composite partitions.

Partition:

```
create table sales_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_jan2000 values less
 than(to_date('01/02/2000','dd/mm/yyyy')), 9 partition sales_feb2000 values
 less than(to_date('01/03/2000','dd/mm/yyyy')), 10 partition sales_mar2000
 values less than(to_date('01/04/2000','dd/mm/yyyy')), 11 partition
 sales_apr2000 values less than(to_date('01/05/2000','dd/mm/yyyy')) 12
 )enable row movement;

Table created.
SQL> insert into sales_range values(1,’Vinit
Wani',5000,to_date('23/02/2000','dd/mm/yyyy')); 1 row created.

SQL> insert into sales_range values(2,'Ajay
ASH',10000,to_date('29/01/2000','dd/mm/yyyy')); 1 row created.

SQL> insert into sales_range values(3,’Aditya
Narkar',50000,to_date('23/03/2000','dd/mm/yyyy')); 1 row created.
```

```
SQL> insert into sales_range values(4,’Omkar

Jadhav',100000,to_date('01/04/2000','dd/mm/yyyy')); 1 row created.


SQL> insert into sales_range values(5,'Shashikumar

Singh',6000,to_date('15/04/2000','dd/mm/yyyy')); 1 row created.


SQL> insert into sales_range values(6,'Mahendra
shign
dhoni',45000,to_date('15/03/2000','dd/mm/yyyy'));

1 row created.


SQL> insert into sales_range values(7,'Ishan

Kishan',66000,to_date('30/04/2000','dd/mm/yyyy')); 1 row created.


SQL> insert into sales_range
values(8,'Ravichandran
Ashivin',25000,to_date('19/02/2000','dd/mm/yyyy'
));

1 row created.


SQL> select * from sales_range;


SALESMAN_ID  SALESMAN_NAME SALES_AMOUNTSALES_DAT

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

      2              Ajay ASH              10000         29-JAN-00

      1              Vinit Wani             5000         23-FEB-00

      8            Ravichandran            25000          19-FEB-00
                      Ashivin
      3             Aditya Narkar          50000         23-MAR-00

      6            Mahendra shign          45000         15-MAR-00
                       dhoni
      4             Omkar Jadhav           100000         01-APR-00

      5          suryakumar yadav            6000         15-APR-00

      7              Ishan Kishan           66000         30-APR-00
```

```
8 rows selected.

SQL> select * from sales_range partition (sales_jan2000);


SALESMAN_ID                          SALESMAN_NAME  SALES_AMOUNT    SALES_DAT

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

    2                Ajay ASH           10000         29-JAN-00


SQL> select * from sales_range partition (sales_feb2000);


        SALESMAN_ID                          SALESMAN_NAME
                               SALES_AMOUNT  SALES_DAT

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

     1                 Vinit Wani             5000        23-FEB-00

     8               Ravichandran            25000        19-FEB-00
                        Ashivin

SQL> select * from sales_range partition (sales_mar2000);


SALESMAN_ID                          SALESMAN_NAME  SALES_AMOUNT    SALES_DAT

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

     3                Aditya Narkar          50000       23-MAR-00

     6          Mahendra shign dhoni        45000       15-MAR-00


SQL> select * from sales_range partition (sales_apr2000);


SALESMAN_ID                          SALESMAN_NAME  SALES_AMOUNT    SALES_DAT

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

     4                Omkar Jadhav         100000       01-APR-00

     5            suryakumar yadav           6000        15-APR-00

     7                Ishan Kishan          66000       30-APR-00
```

```
SQL> create table employee11
  2 (emp_no integer primary
  key, 3 emp_name
  varchar2(20),
  4  salary integer)
  5  partition by
  range(emp_no) 6  (
  7    partition    e1    values    less
  than(105),  8  partition  e2  values
  less than(110) 9 )
 10 ;

Table created.

SQL> insert into employee11 values(101,'kl
rahul',20000); 1 row created.

SQL> insert into employee11 values(102,'shikar
dhawan',19000); 1 row created.

SQL> insert into employee11 values(103,'Rishab
Pant',30000); 1 row created.

SQL> insert into employee11 values(104,'bhuwaneshwar
kumar',15000); 1 row created.

SQL> insert into employee11 values(105,'hardik
pandaya',35000); 1 row created.
```

```
SQL> insert into employee11 values(106,'kurnal
pandaya',12000); 1 row created.

SQL> insert into employee11 values(107,'ravindra
jadeja',50000); 1 row created.

SQL> insert into employee11
values(108,'bumraha',55000); 1 row created.

SQL> select * from employee11;

   EMP_NO        EMP_NAME          SALARY
--------------------------------------------
     101             kl rahul            20000
     102           shikar dhawan          19000
     103            Rishab Pant           30000
     104      bhuwaneshwar kumar          15000
     105          hardik pandaya          35000
     106          kurnal pandaya          12000
     107          ravindra jadeja         50000
     108              bumraha             5500
                                          0

8 rows selected.
```

```
SQL> select * from employee11 partition (e1);


   EMP_NO         EMP_NAME          SALARY
--------------------------------------------
     101              kl rahul            20000
     102           shikar dhawan          19000
     103             Rishab Pant          30000
     104      bhuwaneshwar kumar          15000

SQL> select * from employee11 partition (e2);


   EMP_NO         EMP_NAME          SALARY
--------------------------------------------
     105           hardik pandaya          35000
     106           kurnal pandaya          12000
     107           ravindra jadeja         50000
     108              bumraha              55000
```

```
SQL> create table
 sales_list 2
 (salesman_id number(5),
 3 salesman_name varchar2(30),
 4 sales_state varchar2(20),
 5 sales_amount number(10),
 6 sales_date date)
 7 partition by list (sales_state)
 8 (partition sales_west values ('andheri',
 'borivali'), 9 partition sales_harbur values
 ('vashi','panvel'),
10 partition sales_cental values
('dadar','thane'), 11 partition sales_other
values (default)
12  )
13 enable row movement
14  ;

Table created.


SQL> insert into sales_list values(1,'Vinit
Wani','dadar',25000,to_date('23/02/2000','dd/mm/yyy
y'));
1 row created.


SQL> insert into sales_list values(2,'Ajay
ASH','andheri',35000,to_date('13/08/2000','dd/mm/yyyy')); 1 row created.


SQL> insert into sales_list values(3,'Aditya
Narkar','vashi',15000,to_date('08/06/2000','dd/mm/yyyy')); 1 row created.
```

```
SQL> insert into sales_list values(4,'Omkar
Jadhav','ghatkopar',55000,to_date('17/09/2000','dd/mm/yyy
y'));

1 row created.


SQL> insert into sales_list values(5,'Sanjay

patil','thane',5000,to_date('16/05/2000','dd/mm/yyyy')); 1 row created.


SQL>

SQL> select * from sales_list;


SALESMAN_ID    SALESMAN_NAME    SALES_STATE    SALES_AMOUNT    SALES_DAT

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

      1           Vinit Wani        dadar         25000          23-FEB-
                                                                 00
      2             Ajay ASH        andheri       35000        13-AUG-00

      3          Aditya Narkar       vashi        15000          08-JUN-
                                                                 00
      4          Omkar Jadhav      ghatkopa       15000           17-SEP-
                                       r                          00
      5          Sanjay patil        thane        15000          16-MAY-
                                                                 00


6 rows selected.


SQL> select * from sales_list partition (sales_west);


SALESMAN_ID    SALESMAN_NAME    SALES_STATE    SALES_AMOUNT    SALES_DAT

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

      2             Ajay ASH        andheri       35000        13-AUG-00
```

```
SQL> select * from sales_list partition (sales_harbur);
SALESMAN_ID    SALESMAN_NAME    SALES_STATE   SALES_AMOUNT    SALES_DAT
--------------------------------------------------------------------------
      3              Aditya Narkar           vashi         15000            08-JUN-00

SQL> select * from sales_list partition (sales_other);
SALESMAN_ID    SALESMAN_NAME    SALES_STATE   SALES_AMOUNT    SALES_DAT
--------------------------------------------------------------------------
      4              Omkar Jadhav         ghatkopar       15000             17-SEP-00

SQL> select * from sales_list partition (sales_central);
SALESMAN_ID    SALESMAN_NAME    SALES_STATE   SALES_AMOUNT    SALES_DAT
--------------------------------------------------------------------------
      1             Vinit Wani                dadar        25000            23-FEB-00
      5              Sanjay patil            thane         15000           16-MAY-00

SQL> create table
 sales_hash 2
 (salesman_id number(5),
 3  salesman_name varchar2(30),
 4  sales_amount number(10),
 5  weak_no number(2))
 6  partition by
 hash(salesman_id) 7
 partitions 4
 8 ;

Table created.
```

```
SQL> insert into sales_hash
values(101,'Sam',5000,18); 1 row created.

SQL> insert into sales_hash
values(102,'Ram',50000,50); 1 row created.

SQL> insert into sales_hash
values(103,'Riya',25000,36); 1 row created.

SQL> insert into sales_hash
values(104,'chinmay',35000,49); 1 row created.

SQL> insert into sales_hash
values(105,'Mitesh',45000,17); 1 row created.

SQL> insert into sales_hash
values(106,'Ajay',15000,1); 1 row created.

SQL> insert into sales_hash
values(107,'Nikhil',85000,53); 1 row created.
```

```
SQL> select * from sales_hash;


SALESMAN_ID   SALESMAN_NAME   SALES_AMOUNT   WEAK_NO
------------------------------------------------------------

     104           chinma           35000           49
                     y
     102            Ram             50000           50

     103            Riya            25000           36

     105           Mitesh           45000           17

     107           Nikhil           85000           53

     101            Sam              5000           18

     106            Ajay            15000            1


7 rows selected.
```

2. Implementation of Analytical queries like Roll_UP, CUBE, First, Last , Lead ,Lag,Rank AND Dense Rank

```
SQL>Create table
ssb1 (emp_no
integer, dep_no
integer, bdate date,
salary integer,
comm integer,
job
varchar2(10));



SQL>insert into ssb1
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&jo
b'); Enter value for emp_no: 101
Enter value for dep_no: 10
Enter value for bdate: 12-jan-
82 Enter value for salary:
22000 Enter value for comm:
1000 Enter value for job:
clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(101,10,'12-jan-82',22000,1000,'clerk')
```

```
SQL>insert into ssb1
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&jo
b'); Enter value for emp_no: 102
Enter value for dep_no: 10
Enter value for bdate: 13-feb-
83 Enter value for salary:
33000 Enter value for comm:
2000 Enter value for job:
clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(102,10,'13-feb-83',33000,2000,'clerk')
1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 103
Enter value for dep_no: 10
Enter value for bdate: 14-mar-
84 Enter value for salary:
44000 Enter value for comm:
200
Enter value for job: clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(103,10,'14-mar-84',44000,200,'clerk')
1 row created.
```

```
SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 104
Enter value for dep_no: 20
Enter value for bdate: 15-apr-
87 Enter value for salary:
55000 Enter value for comm:
3000 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(104,20,'15-apr-87',55000,3000,'manager')
1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 105
Enter value for dep_no: 20
Enter value for bdate: 14-jun-
82 Enter value for salary:
38000 Enter value for comm:
4500 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(105,20,'14-jun-82',38000,4500,'manager')
1 row created.
```

```
SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 106
Enter value for dep_no: 20
Enter value for bdate: 15-
aug-88 Enter value for salary:
44000 Enter value for comm:
500  Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(106,20,'15-aug-88',44000,500,'manager')
1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 107
Enter value for dep_no: 10
Enter value for bdate: 31-
dec-81 Enter value for salary:
58000 Enter value for comm:
8000 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(107,10,'31-dec-81',58000,8000,'manager')
1 row created.
```

```
SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 108
Enter value for dep_no: 20
Enter value for bdate: 25-jul-
88 Enter value for salary:
49000 Enter value for
comm: 700 Enter value for
job: clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(108,20,'25-jul-88',49000,700,'clerk')
1 row created.

SQL> select * from ssb1;

   EMP_NO     DEP_NO BDATE    SALARY     COMM       JOB
-------------------------------------------------------------
      101          10 12-JAN-82        22000       1000         clerk
      102          10 13-FEB-83        33000        2000         clerk
      103          10 14-MAR-84        44000         200        clerk
      104          20 15-APR-87         55000       3000    manager
      105          20 14-JUN-82         38000         4500    manager
      106          20 15-AUG-88         44000          500    manager
      107          10 31-DEC-81         58000         8000    manager
      108           20 25-JUL-88         49000           700         clerk

8 rows selected.
```

```
SQL> SELECT
 dep_no,job,count(*),sum(salary) 2 from
 ssb1
 3 group by rollup(dep_no,job);

  DEP_NO     JOB       COUNT(*)       SUM(SALARY)
------------------------------------------------------------------
     10          clerk          3                    99000
     10        manager        1                    58000
     10                         4                  157000
     20          clerk          1                    49000
     20       manager         3                  137000
     20                         4                  186000
                                8                   343000

7 rows selected.

SQL> select
 dep_no,job,sum(salary) 2 from
 ssb1
 3 where dep_no in(10,20)
 4 group by dep_no, rollup(job);

  DEP_NO JOB  SUM(SALARY)
----------------------------------
     10 clerk
                    9900
     0
     10 manager     58000
     10           157000
     20 clerk       49000
```

```
20 manager   137000
20           186000
```

```
6 rows selected.

SQL> SELECT
 dep_no,job,count(*),sum(salary) 2 from
 ssb1
 3 group by job,rollup(dep_no);

   DEP_NO JOB    COUNT(*) SUM(SALARY)
-----------------------------
       10 clerk           3      99000
       20 clerk           1      49000
          clerk           4     148000
       10 manager         1      58000
       20 manager         3
                                 13700
                          0
          manager         4    195000

6 rows selected.

SQL> SELECT
 dep_no,job,count(*),sum(salary) 2 from
 ssb1
 3 group by dep_no, rollup(job);

   DEP_NO JOB    COUNT(*) SUM(SALARY)
-----------------------------
       10 clerk           3      99000
       10 manager         1      58000
       10                 4     157000
       20 clerk           1      49000
```

```
    20 manager        3     137000
```

```
        20                4     186000

  6 rows selected.

  SQL> SELECT
    dep_no,job,count(*),sum(salary) 2 from
    ssb1
    3 group by cube(dep_no,job);

     DEP_NO JOB    COUNT(*) SUM(SALARY)
  -----------------------------
                        8     343000
           clerk           4     148000
           manager         4     195000
        10                4     157000
        10 clerk           3     99000
        10 manager         1     58000
        20                4     186000
        20 clerk           1     49000
        20 manager         3    137000

  9 rows selected.

  SQL> select emp_no,dep_no,salary,comm,
    2 rank() over(partition by dep_no order by salary)as Rank from ssb1;

     EMP_NO  DEP_NO  SALARY   COMM     RANK
  -----------------------------------
       101      10     2200      1000      1
                       0
       102      10     3300      2000      2
                       0
       103       10     4400       200        3
                           0
       107       10     5800      8000        4
                           0
       105       20     3800      4500        1
                           0
       106       20     4400       500        2
                           0
       108       20     4900       700        3
                           0
       104       20     5500      3000        4
                           0


8 rows selected.


SQL> update ssb1
  2  set salary=33000
  3  where emp_no=101;


1 row updated.


SQL> update ssb1
  2  set salary=44000
  3  where emp_no=105;


1 row updated.


SQL> select emp_no,dep_no,salary,comm,
  2 rank() over(partition by dep_no order by salary)as Rank from ssb1;


    EMP_NO  DEP_NO  SALARY   COMM     RANK
------------------------------------

       101       10     3300       100        1
                           0         0
       102       10     3300       200        1
                           0         0
```

103	10	44000	200	3

```
      107       10     5800      8000         4
                          0
      105       20     4400      4500         1
                          0
      106       20     4400       500        1
                          0
      108       20     4900       700        3
                          0
      104       20     5500      3000         4
                          0
```

8 rows selected.

```
SQL> select emp_no,dep_no,salary,comm,
  2 dense_rank() over(partition by dep_no order by salary)as Rank from ssb1;

   EMP_NO  DEP_NO  SALARY   COMM     RANK
------------------------------------------
      101       10     3300      1000         1
                          0
      102       10     3300      2000         1
                          0
      103       10     4400       200        2
                          0
      107       10     5800      8000         3
                          0
      105       20     4400      4500         1
                          0
      106       20     4400       500        1
                          0
      108       20     4900       700        2
                          0
      104       20     5500      3000         3
                          0
```

8 rows selected.

```
SQL> select emp_no,bdate,
  2 lead(bdate,1) over(order by bdate)
```

```
as"next" 3 from ssb1;
```

```
EMP_NO BDATE   next
------------- ------
      107 31-DEC-81 12-JAN-82
      101 12-JAN-82 14-JUN-82
      105 14-JUN-82 13-FEB-83
      102 13-FEB-83 14-MAR-84
      103 14-MAR-84 15-APR-87
      104 15-APR-87 25-JUL-88
      108 25-JUL-88 15-AUG-88
      106 15-AUG-88

8 rows selected.

SQL> select emp_no,bdate,
  2 lead(bdate,1) over(order by bdate)
  as"next" 3 from ssb1 where dep_no=10;

    EMP_NO BDATE            next
------------- ------
      107 31-DEC-81 12-JAN-82
      101 12-JAN-82 13-FEB-83
      102 13-FEB-83 14-MAR-84
      103 14-MAR-84

SQL> select dep_no,salary,
  2 max(salary)keep(DENSE_RANK FIRST ORDER
  BY comm) 3 over(PARTITION BY dep_no)"max"
  4 from ssb1;
```

```
   DEP_NO  SALARY    max
---------------------
       10     5800     44000
                 0
       10     4400     44000
                 0
       10     3300     44000
                 0
       10     3300     44000
                 0
       20     4400     44000
                 0
       20     4400     44000
                 0
       20     4900     44000
                 0
       20     5500     44000
                 0


8 rows selected.


SQL> select * from ssb1;


   EMP_NO  DEP_NO BDATE   SALARY   COMM JOB
------------------- ---------------------
      101       10 12-JAN-82  33000      1000 clerk
      102       10 13-FEB-83 33000      2000 clerk
      103       10 14-MAR-      44000      200 clerk
                84
      104       20 15-APR-     55000    3000 manager
                87
      105       20 14-JUN-82  44000    4500 manager
      106       20 15-AUG-     44000    500 manager
                88
      107       10 31-DEC-     58000    8000 manager
                81
      108       20 25-JUL-88  49000      700 clerk
```

```
8 rows selected.


SQL> select dep_no,salary,
 2 max(salary)keep(DENSE_RANK FIRST ORDER
 BY salary) 3 over(PARTITION BY dep_no)"max"
 4 from ssb1;

  DEP_NO  SALARY    max
---------------------
     10    5800    33000
            0
     10    4400    33000
            0
     10    3300    33000
            0
     10    3300    33000
            0
     20    4400    44000
            0
     20    4400    44000
            0
     20    4900    44000
            0
     20    5500    44000
            0

8 rows selected.

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 ssb1;

  DEP_NO  SALARY    max
---------------------
     10    5800    58000
            0
     10    4400    58000
            0
     10    3300    58000
            0
```

10	33000	58000
20	44000	55000

```
      20     4400     55000
               0
      20     4900     55000
               0
      20     5500     55000
               0


8 rows selected.


SQL> select dep_no,salary,
 2 max(salary)keep(DENSE_RANK LAST ORDER
 BY salary) 3 over(PARTITION BY dep_no)"max"
 4 from ssb1;


  DEP_NO  SALARY    max
---------------------
      10     5800     58000
               0
      10     4400     58000
               0
      10     3300     58000
               0
      10     3300     58000
               0
      20     4400     55000
               0
      20     4400     55000
               0
      20     4900     55000
               0
      20     5500     55000
               0


8 rows selected.


SQL> select dep_no,salary,
 2 max(salary)keep(DENSE_RANK LAST ORDER BY
 salary desc) 3 over(PARTITION BY dep_no)"max"
```

```
4 from ssb1;
```

```
   DEP_NO  SALARY    max
---------------------
      10     5800    33000
                0
      10     4400    33000
                0
      10     3300    33000
                0
      10     3300    33000
                0
      20     4400    44000
                0
      20     4400    44000
                0
      20     4900    44000
                0
      20     5500    44000
                0


8 rows selected.
```

```
Create table
ssb1 (emp_no
integer, dep_no
integer, bdate
date, salary
integer, comm
integer, job
varchar2(10));



insert into ssb1
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&jo
b'); Enter value for emp_no: 101
Enter value for dep_no: 10
Enter value for bdate: 12-jan-
```

```
82 Enter value for salary:
22000 Enter value for comm:
1000
```

```
Enter value for job: clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(101,10,'12-jan-82',22000,1000,'clerk')

insert into ssb1
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&jo
b'); Enter value for emp_no: 102
Enter value for dep_no: 10
Enter value for bdate: 13-feb-
83 Enter value for salary:
33000 Enter value for comm:
2000 Enter value for job:
clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(102,10,'13-feb-83',33000,2000,'clerk')

1 row created.

insert into ssb1
  2 2 values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 103
Enter value for dep_no: 10
Enter value for bdate: 14-mar-
84 Enter value for salary:
44000 Enter value for comm:
200
Enter value for job: clerk
old 2: 2 values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: 2 values(103,10,'14-mar-84',44000,200,'clerk')
  2 values(103,10,'14-mar-84',44000,200,'clerk')
  *
```

```
ERROR at line 2:
ORA-00926: missing VALUES keyword



SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 103
Enter value for dep_no: 10
Enter value for bdate: 14-mar-
84 Enter value for salary:
44000 Enter value for comm:
200
Enter value for job: clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(103,10,'14-mar-84',44000,200,'clerk')

1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 104
Enter value for dep_no: 20
Enter value for bdate: 15-apr-
87 Enter value for salary:
55000 Enter value for comm:
3000 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(104,20,'15-apr-87',55000,3000,'manager')
```

```
1 row created.
```

```
SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 105
Enter value for dep_no: 20
Enter value for bdate: 14-jun-
82 Enter value for salary:
38000 Enter value for comm:
4500 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(105,20,'14-jun-82',38000,4500,'manager')

1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 106
Enter value for dep_no: 20
Enter value for bdate: 15-
aug-88 Enter value for salary:
44000 Enter value for comm:
500  Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(106,20,'15-aug-88',44000,500,'manager')

1 row created.
```

```
SQL> insert into ssb1
  2 values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
```

```
Enter value for emp_no: 107
Enter value for dep_no: 10
Enter value for bdate: 31-
dec-81 Enter value for salary:
58000 Enter value for comm:
8000 Enter value for job:
manager
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(107,10,'31-dec-81',58000,8000,'manager')

1 row created.

SQL> insert into ssb1
  2
values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job');
Enter value for emp_no: 108
Enter value for dep_no: 20
Enter value for bdate: 25-jul-
88 Enter value for salary:
49000 Enter value for
comm: 700 Enter value for
job: clerk
old 2: values(&emp_no,&dep_no,'&bdate',&salary,&comm,'&job')
new 2: values(108,20,'25-jul-88',49000,700,'clerk')

1 row created.

SQL> select * from ssb1;

   EMP_NO  DEP_NO BDATE   SALARY   COMM JOB
------------------ --------------------
```

```
       101        10 12-JAN-82   22000      1000 clerk
```

```
      102       10 13-FEB-83 33000      2000 clerk

      103       10 14-MAR-      44000      200 clerk
                84
      104       20 15-APR-     55000    3000 manager
                87
      105       20 14-JUN-82  38000    4500 manager

      106       20 15-AUG-      44000     500 manager
                88
      107       10 31-DEC-     58000    8000 manager
                81
      108       20 25-JUL-88  49000      700 clerk


8 rows selected.


SQL> SELECT
 dep_no,job,count(*),sum(salary) 2 from
 ssb1
 3 group by rollup(dep_no,job);


  DEP_NO JOB    COUNT(*) SUM(SALARY)
----------------------------
     10 clerk           3      99000
     10 manager         1      58000
     10                4     157000
     20 clerk           1      49000
     20 manager         3     137000
     20                4     186000
                     8     343000

7 rows selected.


SQL> select
 dep_no,job,sum(salary) 2 from
```

```
ssb1
3 where dep_no in(10,20)
```

```
  4 group by dep_no, rollup(job);

   DEP_NO JOB   SUM(SALARY)
---------------------------------------- -
      10 clerk
                        9900
      0
      10 manager      58000
      10             157000
      20 clerk        49000
      20 manager     137000
      20             186000

6 rows selected.

SQL> SELECT
  dep_no,job,count(*),sum(salary) 2 from
  ssb1
  3 group by job,rollup(dep_no);

   DEP_NO JOB     COUNT(*) SUM(SALARY)
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
      10 clerk           3     99000
      20 clerk           1     49000
         clerk           4    148000
      10 manager         1     58000
      20 manager         3
                                13700
                         0
         manager         4    195000

6 rows selected.
```

```
SQL> SELECT dep_no,job,count(*),sum(salary)
```

```
  2  from ssb1
  3  group by dep_no, rollup(job);

    DEP_NO JOB     COUNT(*) SUM(SALARY)
-----------------------------
        10 clerk            3      99000
        10 manager          1      58000
        10                  4     157000
        20 clerk            1      49000
        20 manager          3     137000
        20                  4     186000

6 rows selected.

SQL> SELECT
  dep_no,job,count(*),sum(salary) 2 from
  ssb1
  3 group by cube(dep_no,job);

    DEP_NO JOB     COUNT(*) SUM(SALARY)
-----------------------------
                         8     343000
           clerk            4     148000
           manager          4     195000
        10                  4     157000
        10 clerk            3      99000
        10 manager          1      58000
        20                  4     186000
        20 clerk            1      49000
        20 manager          3     137000
```

```
9 rows selected.


SQL> select emp_no,dep_no,salary,comm,
  2 rank() over(partition by dep_no order by salary)as Rank from ssb1;


    EMP_NO  DEP_NO  SALARY   COMM     RANK
----------------------------------
       101      10     2200      100       1
                          0        0
       102      10     3300      200       2
                          0        0
       103      10     4400      200      3
                          0
       107      10     5800      800       4
                          0        0
       105      20     3800      450       1
                          0        0
       106      20     4400      500      2
                          0
       108      20     4900      700      3
                          0
       104      20     5500      300       4
                          0        0


8 rows selected.


SQL> update ssb1
  2  set salary=33000
  3  where emp_no=101;


1 row updated.


SQL> update ssb1
  2  set salary=44000
  3  where emp_no=105;
```

```
1 row updated.


SQL> select emp_no,dep_no,salary,comm,
  2 rank() over(partition by dep_no order by salary)as Rank from ssb1;


    EMP_NO  DEP_NO  SALARY   COMM     RANK
----------------------------------
       101      10    3300     1000       1
                         0
       102      10    3300     2000       1
                         0
       103      10    4400      200      3
                         0
       107      10    5800     8000       4
                         0
       105      20    4400     4500       1
                         0
       106      20    4400      500      1
                         0
       108      20    4900      700      3
                         0
       104      20    5500     3000       4
                         0


8 rows selected.


SQL> select emp_no,dep_no,salary,comm,
  2 dense_rank() over(partition by dep_no order by salary)as Rank from ssb1;


    EMP_NO  DEP_NO  SALARY   COMM     RANK
----------------------------------
       101      10    3300      100       1
                         0        0
       102      10    3300      200       1
                         0        0
       103      10    4400      200      2
                         0
       107      10    5800      800       3
```

		0	0	
105	20	44000	4500	1
106	20	44000	500	1

```
       108       20    49000      700        2
       104       20    55000     3000        3

8 rows selected.

SQL> select emp_no,bdate,
  2 lead(bdate,1) over(order by bdate)
  as"next" 3 from ssb1;

    EMP_NO BDATE           next
------------- ------
       107 31-DEC-81 12-JAN-82
       101 12-JAN-82 14-JUN-82
       105 14-JUN-82 13-FEB-83
       102 13-FEB-83 14-MAR-84
       103 14-MAR-84 15-APR-87
       104 15-APR-87 25-JUL-88
       108 25-JUL-88 15-AUG-88
       106 15-AUG-88

8 rows selected.

SQL> select emp_no,bdate,
  2 lead(bdate,1) over(order by bdate)
  as"next" 3 from ssb1 where dep_no=10;

    EMP_NO BDATE           next
------------- ------
       107 31-DEC-81 12-JAN-82
```

```
      101 12-JAN-82 13-FEB-83

      102 13-FEB-83 14-MAR-84

      103 14-MAR-84


SQL> select dep_no,salary,

  2 max(salary)keep(DENSE_RANK FIRST ORDER

  BY comm) 3 over(PARTITION BY dep_no)"max"

  4 from ssb1;


    DEP_NO  SALARY    max

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

       10     5800     44000
                0
       10     4400     44000
                0
       10     3300     44000
                0
       10     3300     44000
                0
       20     4400     44000
                0
       20     4400     44000
                0
       20     4900     44000
                0
       20     5500     44000
                0


8 rows selected.


SQL> select * from ssb1;


    EMP_NO  DEP_NO BDATE   SALARY   COMM JOB

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

      101       10 12-JAN-82  33000    1000 clerk

      102       10 13-FEB-83  33000    2000 clerk
```

```
       103         10 14-MAR-        44000       200 clerk
                      84
```

```
      104    20 15-APR-87    55000    3000 manager

      105     20 14-JUN-82    44000    4500 manager

      106     20 15-AUG-88    44000     500 manager

      107    10 31-DEC-81    58000    8000 manager

      108    20 25-JUL-88     49000      700 clerk


8 rows selected.


SQL> select dep_no,salary,

  2 max(salary)keep(DENSE_RANK FIRST ORDER

  BY salary) 3 over(PARTITION BY dep_no)"max"

  4 from ssb1;


    DEP_NO  SALARY    max

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

        10     5800     33000
                 0
        10     4400     33000
                 0
        10     3300     33000
                 0
        10     3300     33000
                 0
        20     4400     44000
                 0
        20     4400     44000
                 0
        20     4900     44000
                 0
        20     5500     44000
                 0


8 rows selected.


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 ssb1;


    DEP_NO  SALARY    max
---------------------
       10      5800     58000
                 0
       10      4400     58000
                 0
       10      3300     58000
                 0
       10      3300     58000
                 0
       20      4400     55000
                 0
       20      4400     55000
                 0
       20      4900     55000
                 0
       20      5500     55000
                 0


8 rows selected.


SQL> select dep_no,salary,
  2 max(salary)keep(DENSE_RANK LAST ORDER
  BY salary) 3 over(PARTITION BY dep_no)"max"
  4 from ssb1;


    DEP_NO  SALARY    max
---------------------
       10      5800     58000
                 0
       10      4400     58000
                 0
       10      3300     58000
                 0
       10      3300     58000
                 0
```

20	44000	55000
20	44000	55000
20	49000	55000

```
        20     55000     55000


8 rows selected.


SQL> select dep_no,salary,
  2 max(salary)keep(DENSE_RANK LAST ORDER BY
  salary desc) 3 over(PARTITION BY dep_no)"max"
  4 from ssb1;


    DEP_NO  SALARY    max
---------------------
        10     5800     33000
                0
        10     4400     33000
                0
        10     3300     33000
                0
        10     3300     33000
                0
        20     4400     44000
                0
        20     4400     44000
                0
        20     4900     44000
                0
        20     5500     44000
                0


8 rows selected.
```

3.Implementation of ORDBMS concepts like ADT(Abstract Data Types), Reference

Inheritance

Referencing and Dereference

```
SQL> create or replace type ANIMAL_TY as
  object 2 (Breed varchar2(25),
  3  Name varchar2(25),
  4  BirthDate DATE);
  5 /

Type created.

SQL> create table ANIMAL of ANIMAL_TY;

Table created.

SQL> insert into ANIMAL VALUES(
  2 ANIMAL_TY('MULE','FRANCES','01-APR-02'));

1 row created.

SQL> insert into ANIMAL VALUES(
  2 ANIMAL_TY('DOG','BENJI','03-APR-01'));

1 row created.
```

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


REF(A)

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

00002802090495375121ED4FABA08F2899F8B0D56CE3210848E0AE41ABA97502CEB5282C6
6010003

460000


000028020994132603ADF248ECA8DB91CBD1CA91D4E3210848E0AE41ABA97502CEB5282
C66010003

460001



SQL> create table KEEPER

  2  (KeeperName varchar2(25),

  3  AnimalKept REF ANIMAL_TY);


Table created.


SQL> describe KEEPER;

 Name                              Null?  Type

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

 KEEPERNAME                                VARCHAR2(25)

 ANIMALKEPT                               REF OF ANIMAL_TY


SQL> insert into

  KEEPER 2 select

  'CATHERINE',

  3  REF(A)

  4  from ANIMAL A

  5  where Name='BENJI';
```

```
1 row created.

SQL> select * from KEEPER;

KEEPERNAME
------------------
ANIMALKEPT
----------------------------------------------------------
CATHERINE
000022020894132603ADF248ECA8DB91CBD1CA91D4E3210848E0AE41ABA97502
CEB5282C66



SQL> set describe
depth 2; SQL> describe
KEEPER;
 Name                          Null?  Type
 ------------------------------ --------------------------
 KEEPERNAME                              VARCHAR2(25)
 ANIMALKEPT                             REF OF
   ANIMAL_TY BREED           VARCHAR2(25)
   NAME                              VARCHAR2(25)
   BIRTHDATE                            DATE



SQL> select
  KeeperName,DEREF(K.AnimalKept) 2
  from KEEPER K;

KEEPERNAME
```

-----------------.--

```
DEREF(K.ANIMALKEPT)(BREED, NAME, BIRTHDATE)
--------------------------------------------------
CATHERINE
ANIMAL_TY('DOG', 'BENJI', '03-APR-01')
```

Inheritance

```
SQL> create or replace type PERSON_TY
 as object 2 (ssn number,
 3 Name varchar2(25),
 4 gender char(1),
 5 static function show_super(person_obj in person_ty) return
 varchar2, 6 member function show return varchar2)
 7 commit
 8 /

Type created.

SQL> commit;

Commit complete.

SQL> create or replace type TEACHER_TY UNDER
 PERSON_TY 2 (d_of_joining DATE,
 3 salary number(7,2),
 4 courses_taught varchar2(50),
 5 OVERRIDING member function show return
 varchar2); 6  /

Type created.

SQL> create or replace type STUDENT_TY UNDER
 PERSON_TY 2 (grade char(1),
 3 yr_of_comp varchar2(20),
 4 courses_taken varchar2(50),
 5 OVERRIDING member function show return
 varchar2); 6  /


Type created.
SQL> create type body PERSON_TY as
 2 --static function that can be called by subtypes
 3 static function show_super(person_obj in person_ty) return
 varchar2 is 4  begin
 5 return
'SSN:'||TO_CHAR(person_obj.ssn)||',Name:'||person_obj.Name||',Gender:'||person_obj.gen
der;
 6 end;
 7 --function that can be overriden by
 subtypes 8 member function show return
 varchar2 is
 9 begin
10 return person_ty.show_super(SELF);
11 end;
12 end;
13 /


Type body created.


SQL> create type body STUDENT_TY as
 2 OVERRIDING member function show return
 varchar2 is 3 begin
 4 return person_ty.show_super(SELF)||'--
Grade:'||grade||',Yr_Of_Comp:'||yr_of_comp||',Courses Taken:'||courses_taken;
 5 end;
 6 end;
 7 /


Type body created.


SQL> create or replace type body TEACHER_TY as
 2 OVERRIDING member function show return
 varchar2 is 3 begin
 4 return person_ty.show_super(SELF)||'--
Date_of_Joining:'||d_of_joining||',Salary:'||TO_CHAR(salary
)||
 5 ',Courses Taught:'||courses_taught;
 6 end;
 7 end;
 8 /


Type body created.
```

Create an Object Table

```
SQL> create table person_obj_table of PERSON_TY;


Table created.
```

Insert Data into Object Table

```
SQL> insert into person_obj_table values(PERSON_TY(1,'Ajay','M'));


1 row created.


SQL> insert into person_obj_table values(PERSON_TY(2,'Rushi','M'));


1 row created.
```

Inserting into TEACHER_TY obj

```
SQL> insert into person_obj_table values(TEACHER_TY(1,'Aditya','M','20 JAN
2022',20000,'CG,DS'));


1 row created.
```

Selecting Data from Object Table

```
SQL> select p.show() from person_obj_table p;


P.SHOW()
--------------------------------------------------
SSN:1,Name:Ajay,Gender:
M
SSN:2,Name:Rushi,Gende
r:M
SSN:1,Name:Aditya,Gender:M--Date_of_Joining: 20-JAN-22,Salary:20000,Courses Taught
:CG,DS
```

5. Implementation of ETL transformation with Pentaho

Open SQL Command prompt

Enter user-name:

Ajay Enter

password:

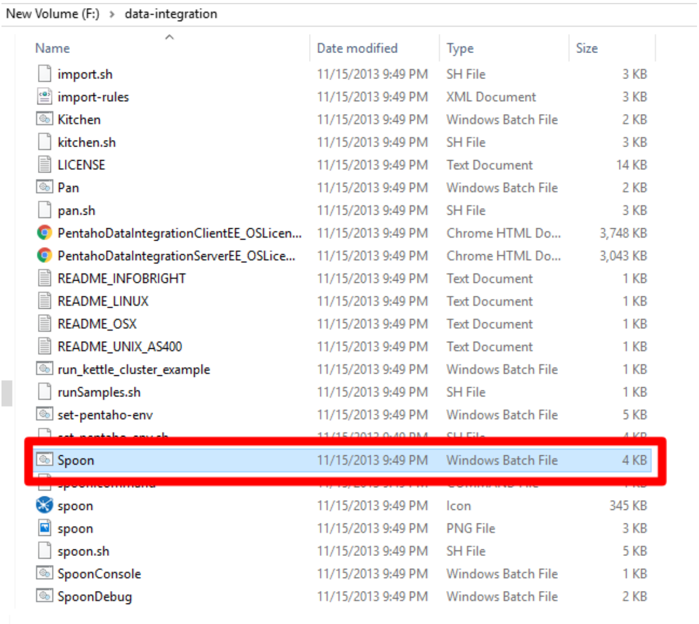
Connected to:

Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Production

With the Partitioning, OLAP, Data Mining and Real Application Testing options

Open spoon from F drive

Double click to spoon:

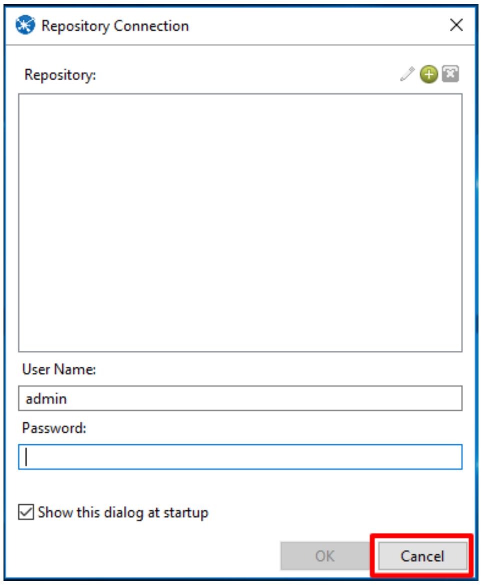


New Volume (F:) > data-integration

Name	Date modified	Type	Size
import.sh	11/15/2013 9:49 PM	SH File	3 KB
import-rules	11/15/2013 9:49 PM	XML Document	3 KB
Kitchen	11/15/2013 9:49 PM	Windows Batch File	2 KB
kitchen.sh	11/15/2013 9:49 PM	SH File	3 KB
LICENSE	11/15/2013 9:49 PM	Text Document	14 KB
Pan	11/15/2013 9:49 PM	Windows Batch File	2 KB
pan.sh	11/15/2013 9:49 PM	SH File	3 KB
PentahoDataIntegrationClientEE_OSLicen...	11/15/2013 9:49 PM	Chrome HTML Do...	3,748 KB
PentahoDataIntegrationServerEE_OSLice...	11/15/2013 9:49 PM	Chrome HTML Do...	3,043 KB
README_INFOBRIGHT	11/15/2013 9:49 PM	Text Document	1 KB
README_LINUX	11/15/2013 9:49 PM	Text Document	1 KB
README_OSX	11/15/2013 9:49 PM	Text Document	1 KB
README_UNIX_AS400	11/15/2013 9:49 PM	Text Document	1 KB
run_kettle_cluster_example	11/15/2013 9:49 PM	Windows Batch File	1 KB
runSamples.sh	11/15/2013 9:49 PM	SH File	1 KB
set-pentaho-env	11/15/2013 9:49 PM	Windows Batch File	5 KB
set-pentaho-env.sh	11/15/2013 9:49 PM	SH File	4 KB
Spoon	11/15/2013 9:49 PM	Windows Batch File	4 KB
SpoonCommand	11/15/2013 9:49 PM	COMMAND File	1 KB
spoon	11/15/2013 9:49 PM	Icon	345 KB
spoon	11/15/2013 9:49 PM	PNG File	3 KB
spoon.sh	11/15/2013 9:49 PM	SH File	5 KB
SpoonConsole	11/15/2013 9:49 PM	Windows Batch File	1 KB
SpoonDebug	11/15/2013 9:49 PM	Windows Batch File	2 KB

A window will popup

Click on cancel:

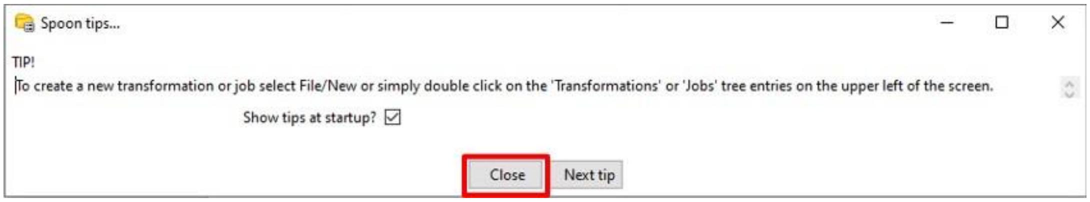


User Name: admin

Password:

☑ Show this dialog at startup

Click Close:

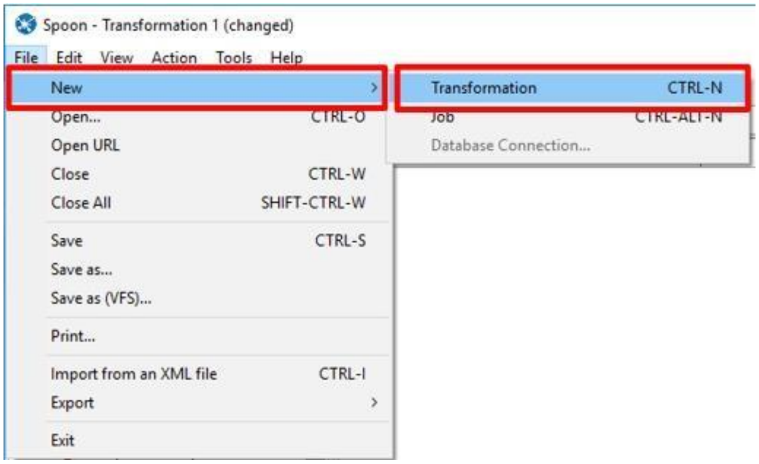


TIP!

To create a new transformation or job select File/New or simply double click on the 'Transformations' or 'Jobs' tree entries on the upper left of the screen.

Show tips at startup? ☑

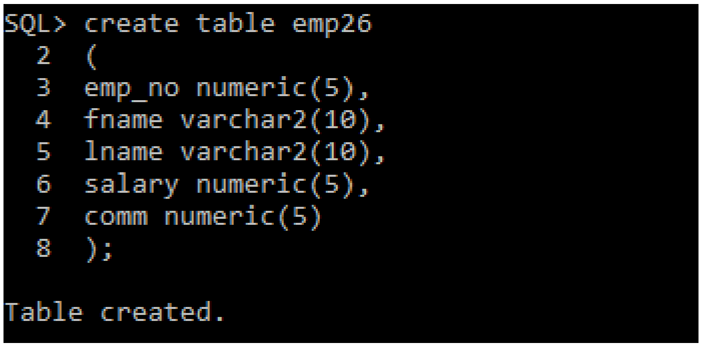
For Creating New Transformation:



Copy data from Source (Table/Excel/ Oracle) and store it to Target

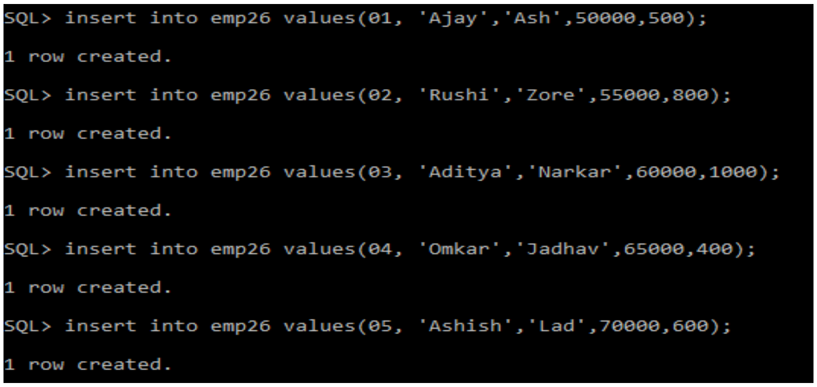
Step 1:

Create table in SQL Plus Command Prompt



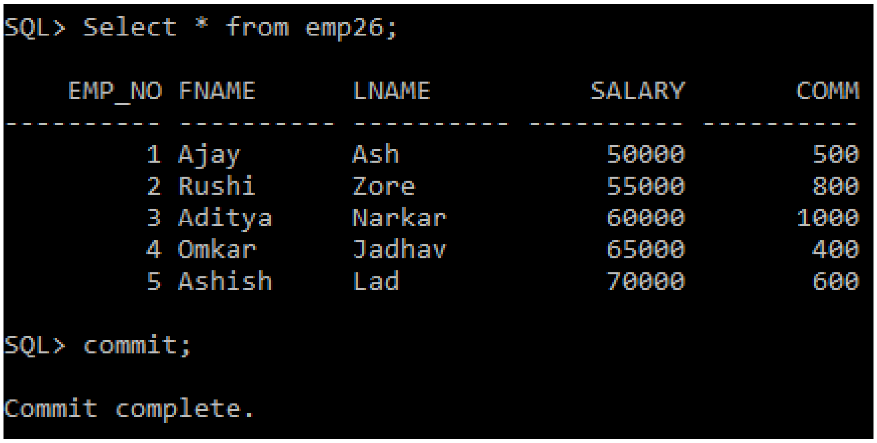
```
SQL> create table emp26
  2  (
  3  emp_no numeric(5),
  4  fname varchar2(10),
  5  lname varchar2(10),
  6  salary numeric(5),
  7  comm numeric(5)
  8  );

Table created.
```

Insert Values Into the tables:



```
SQL> insert into emp26 values(01, 'Ajay','Ash',50000,500);

1 row created.

SQL> insert into emp26 values(02, 'Rushi','Zore',55000,800);

1 row created.

SQL> insert into emp26 values(03, 'Aditya','Narkar',60000,1000);

1 row created.

SQL> insert into emp26 values(04, 'Omkar','Jadhav',65000,400);

1 row created.

SQL> insert into emp26 values(05, 'Ashish','Lad',70000,600);

1 row created.
```

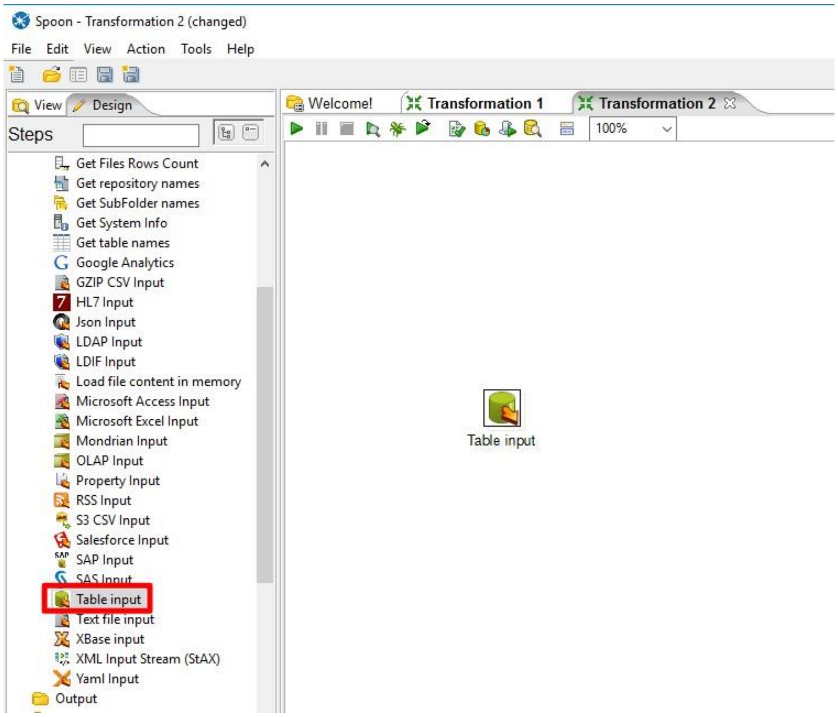
Show Table Values And Commit :



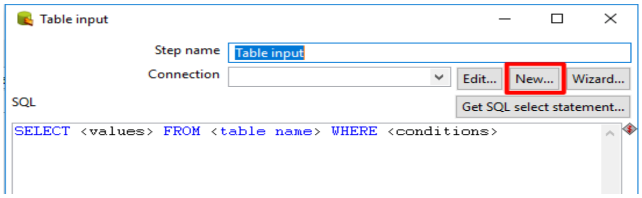
```
SQL> Select * from emp26;

    EMP_NO FNAME      LNAME          SALARY       COMM
---------- ---------- ---------- ---------- ----------
         1 Ajay       Ash             50000        500
         2 Rushi      Zore            55000        800
         3 Aditya     Narkar          60000       1000
         4 Omkar      Jadhav          65000        400
         5 Ashish     Lad             70000        600

SQL> commit;

Commit complete.
```

Step 2:

Open Pentaho Then Select table input and drag:



Double click on table input :

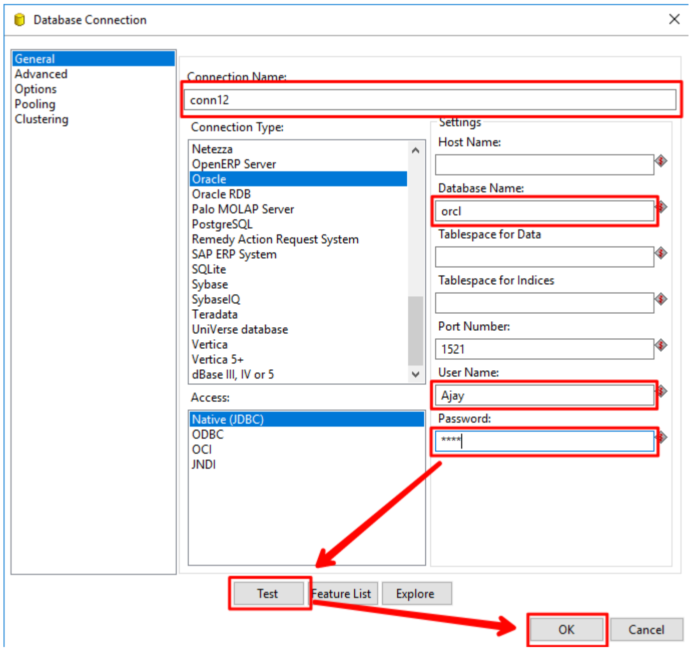


Step name: Table input

Connection:

SQL

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


Database Connection

Connection Name: conn12

Connection Type: Oracle

Access: Native (JDBC)

Settings

Host Name:

Database Name: orcl

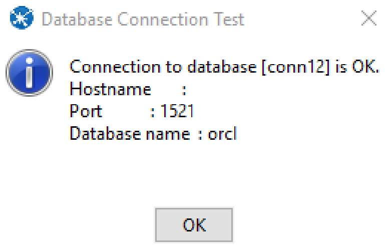
Tablespace for Data

Tablespace for Indices

Port Number: 1521

User Name: Ajay

Password: ****


Database Connection Test

Connection to database [conn12] is OK.
Hostname :
Port : 1521
Database name : orcl

This message will come which show there is no error till now Select the table which you had made in SQL command prompt

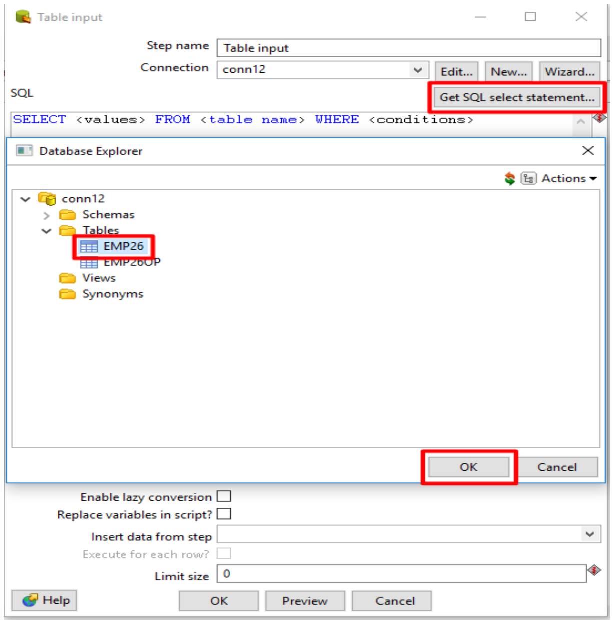


Table input

Step name: Table input

Connection: conn12

SQL

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

Database Explorer

- conn12
  - Schemas
  - Tables
    - EMP26
    - EMP26OP
  - Views
  - Synonyms

Enable lazy conversion □

Replace variables in script? □

Insert data from step

Execute for each row? □

Limit size 0

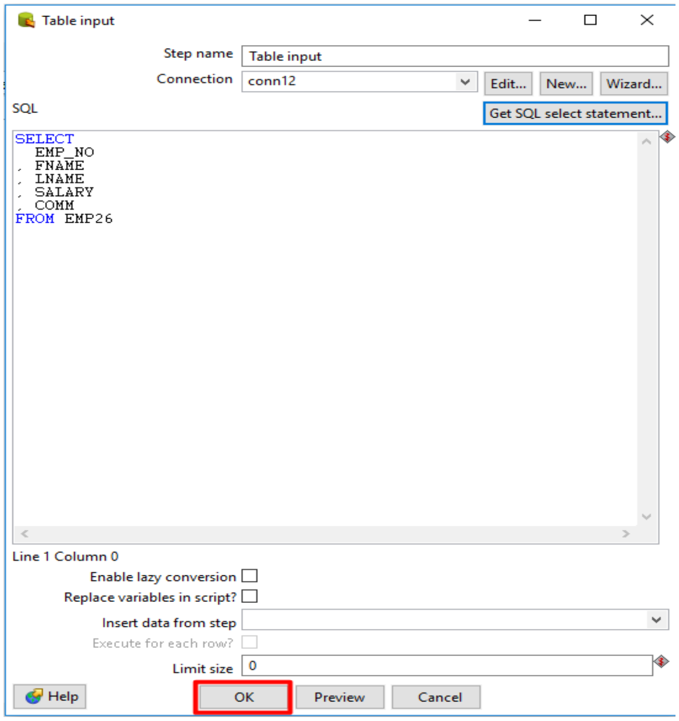


Table input

Step name: Table input

Connection: conn12

SQL

```
SELECT
   EMP_NO
,  FNAME
,  LNAME
,  SALARY
,  COMM
FROM EMP26
```

Line 1 Column 0

Enable lazy conversion □

Replace variables in script? □

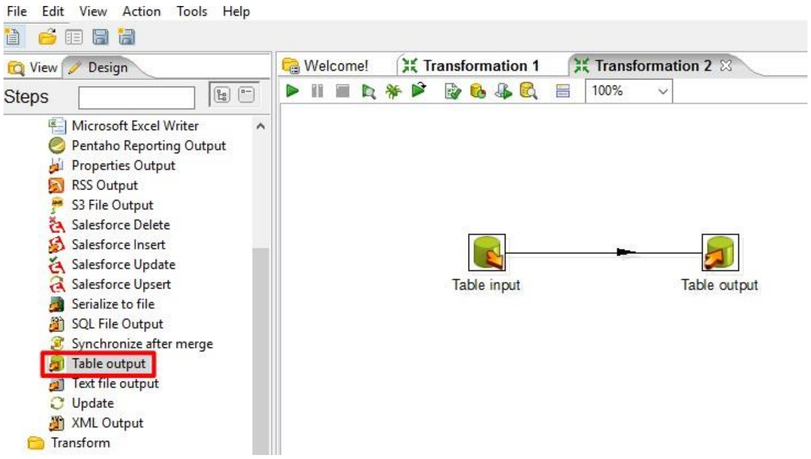
Insert data from step:

Execute for each row? □

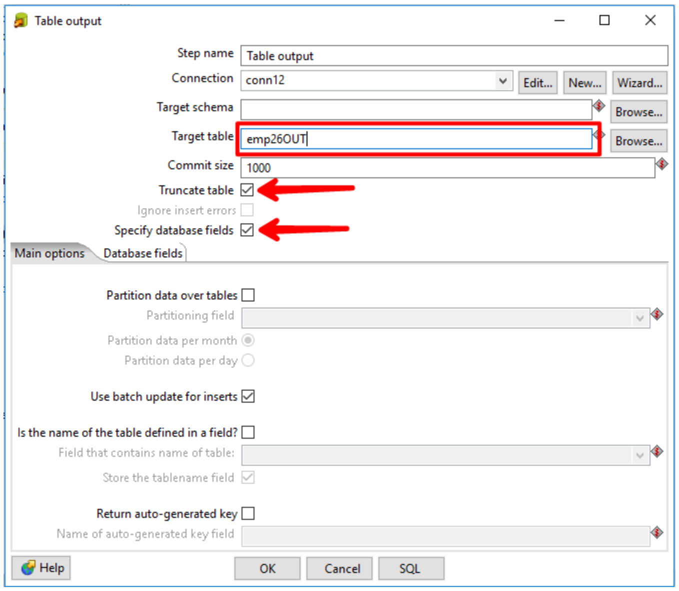
Limit size: 0

Step 3:

Select on table output and drag It:



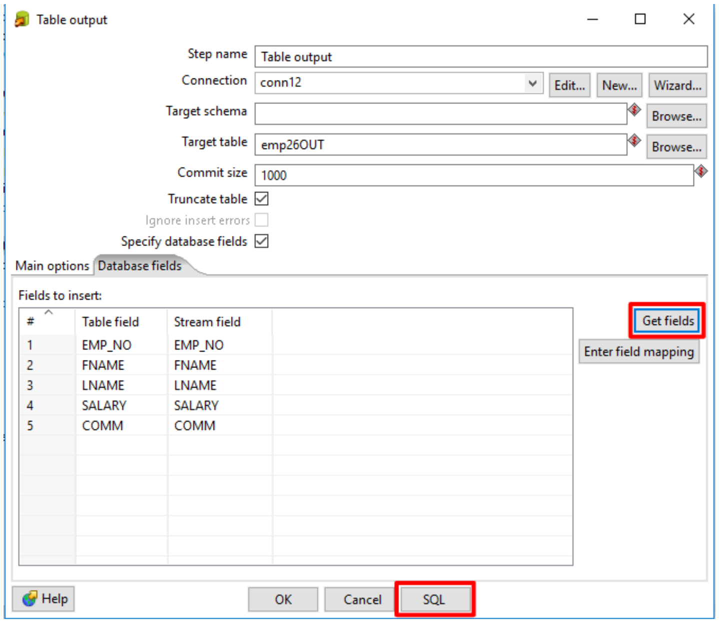
Double Click On Table Output:



Step name: Table output
Connection: conn12
Target schema:
Target table: emp26OUT
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

Please check Truncate table and Specify database fields Go On Database Field And Click on Get fields :



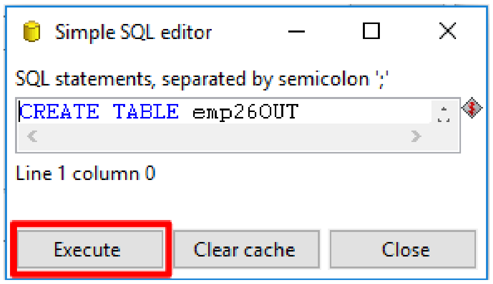
Step name: Table output
Connection: conn12
Target schema:
Target table: emp26OUT
Commit size: 1000
Truncate table ☑
Ignore insert errors □
Specify database fields ☑

Fields to insert:

#	Table field	Stream field
1	EMP_NO	EMP_NO
2	FNAME	FNAME
3	LNAME	LNAME
4	SALARY	SALARY
5	COMM	COMM

Step 4:

Execute The Command:

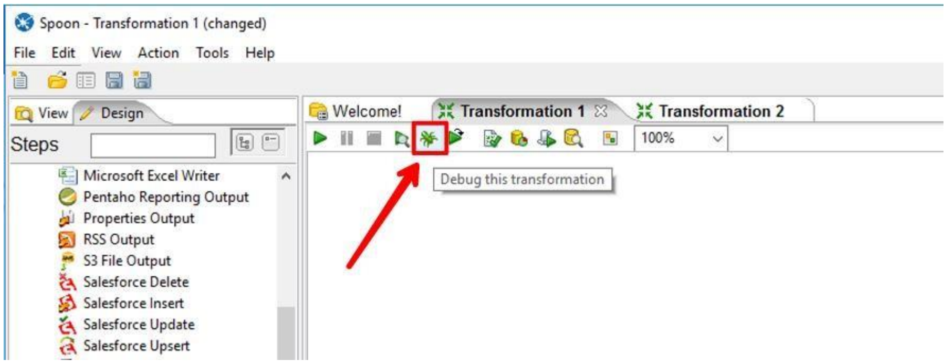


SQL statements, separated by semicolon ';'

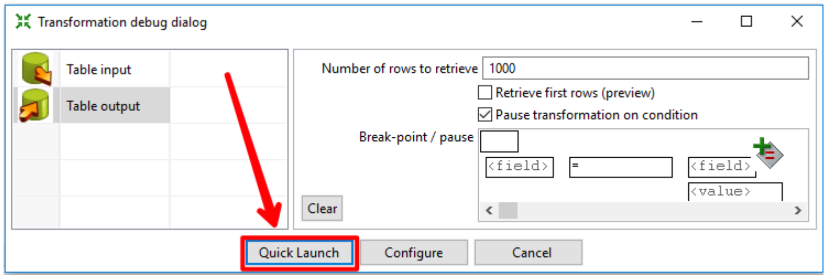
```
CREATE TABLE emp26OUT
```

Line 1 column 0

Click On Debug :



Click On Quick Launch :



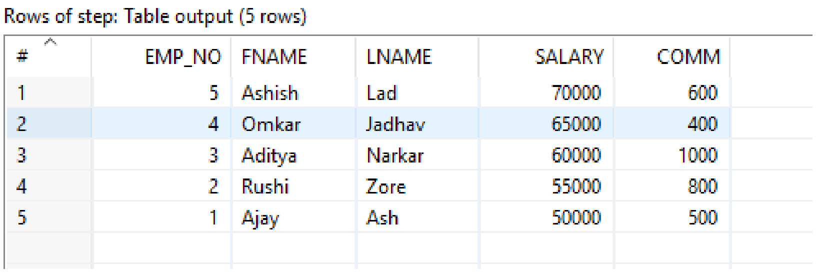
Number of rows to retrieve 1000

☐ Retrieve first rows (preview)

☑ Pause transformation on condition

Break-point / pause

Display Of Output Table emp26OP :

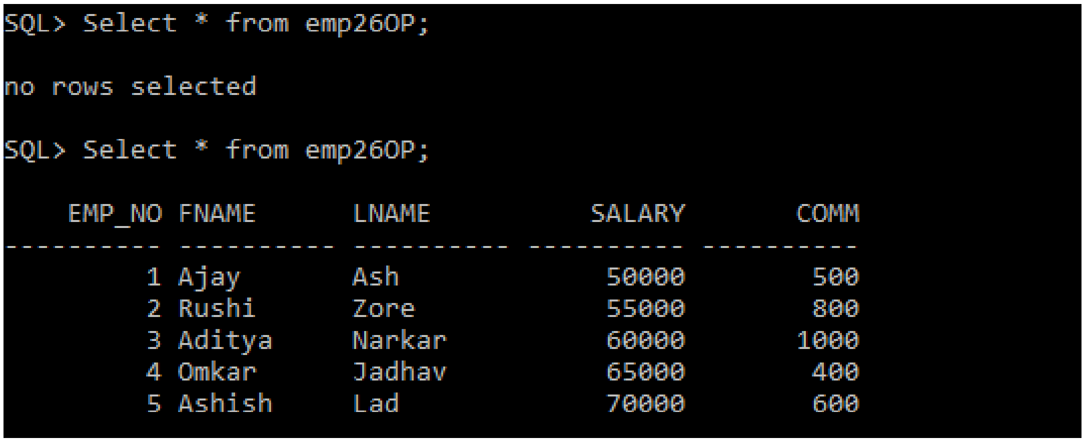


Rows of step: Table output (5 rows)

#	EMP_NO	FNAME	LNAME	SALARY	COMM
1	5	Ashish	Lad	70000	600
2	4	Omkar	Jadhav	65000	400
3	3	Aditya	Narkar	60000	1000
4	2	Rushi	Zore	55000	800
5	1	Ajay	Ash	50000	500

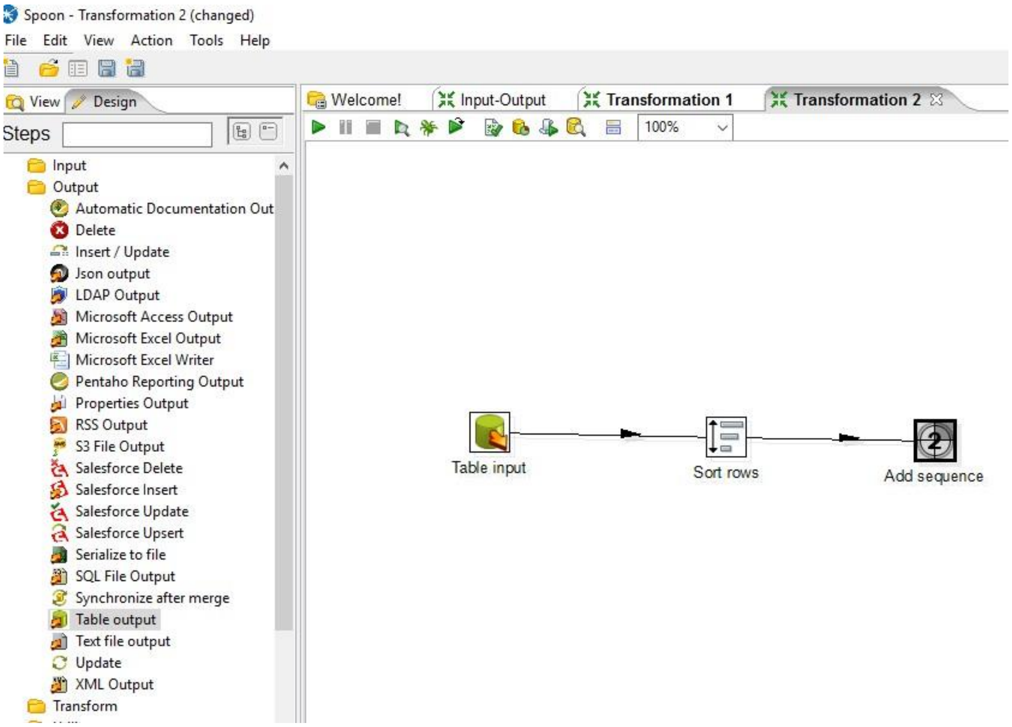
Step : 5

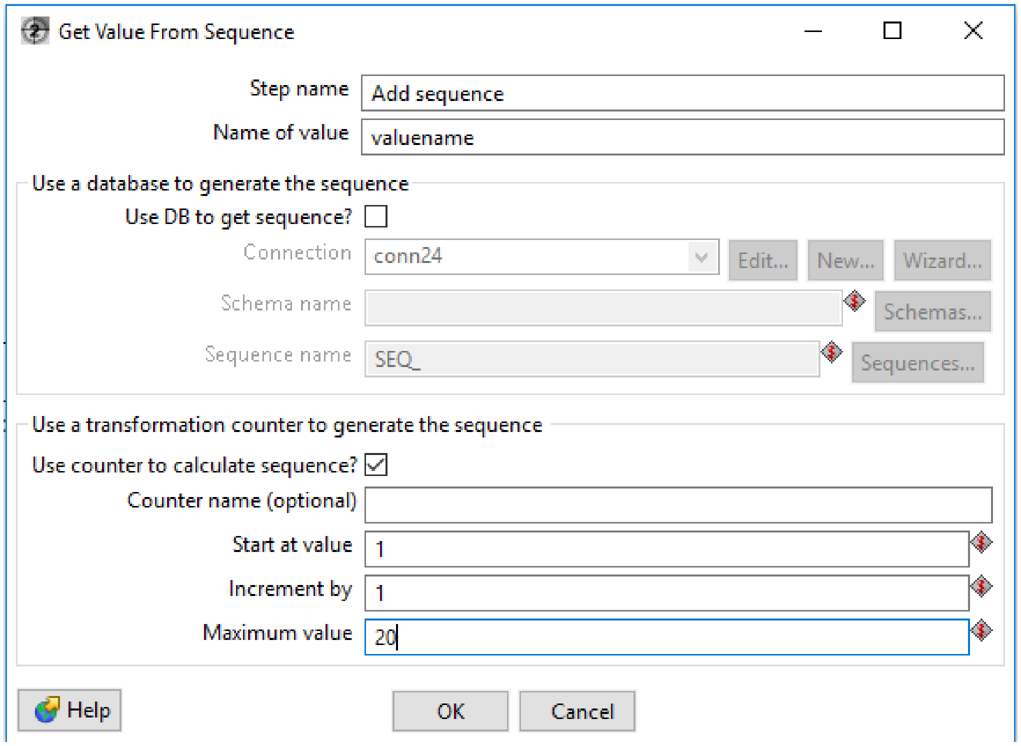
Display Of Output Table emp26OP In SQL Plus :



```
SQL> Select * from emp26OP;

no rows selected

SQL> Select * from emp26OP;

    EMP_NO FNAME      LNAME          SALARY       COMM
---------- ---------- ---------- ---------- ----------
         1 Ajay       Ash             50000        500
         2 Rushi      Zore            55000        800
         3 Aditya     Narkar          60000       1000
         4 Omkar      Jadhav          65000        400
         5 Ashish     Lad             70000        600
```

Adding Sequence:





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: conn24

Schema name:

Sequence name: SEQ_

Use a transformation counter to generate the sequence

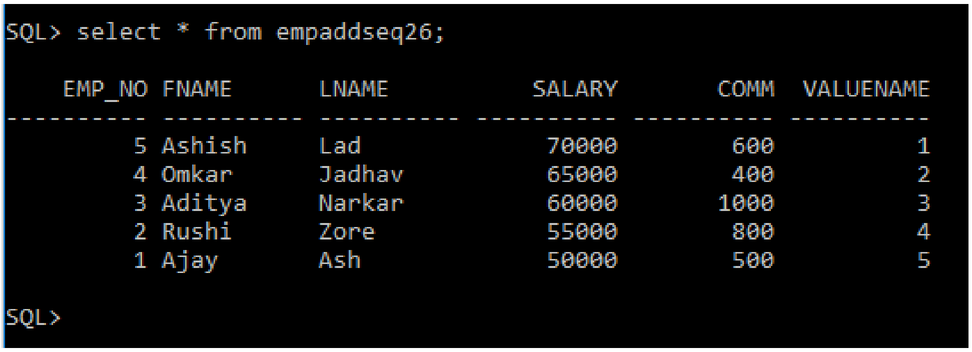
Use counter to calculate sequence? ☑

Counter name (optional):

Start at value: 1

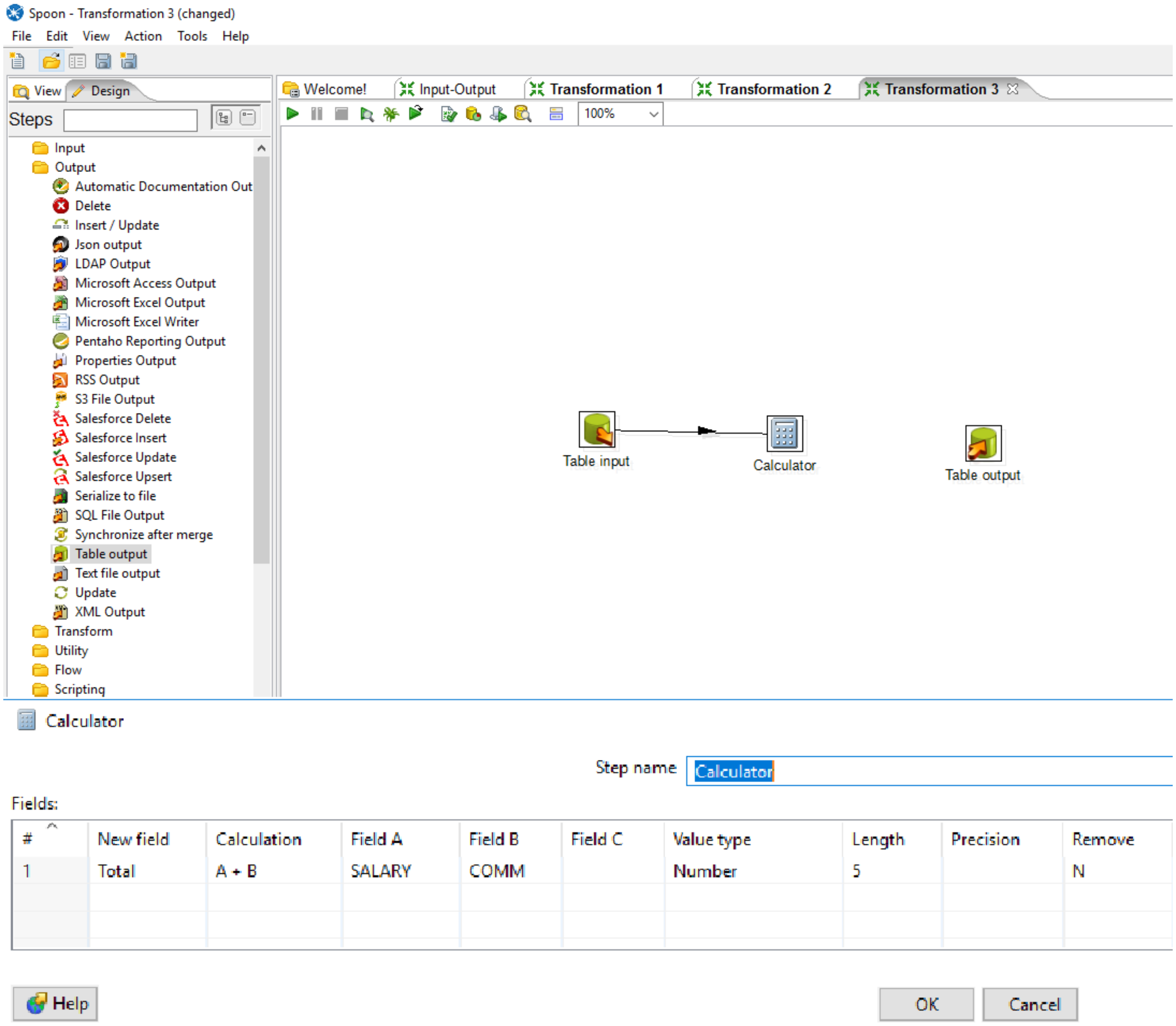
Increment by: 1

Maximum value: 20



```
SQL> select * from empaddseq26;

    EMP_NO FNAME      LNAME          SALARY       COMM  VALUENAME
---------- ---------- ---------- ---------- ---------- ----------
         5 Ashish     Lad             70000        600          1
         4 Omkar      Jadhav          65000        400          2
         3 Aditya     Narkar          60000       1000          3
         2 Rushi      Zore            55000        800          4
         1 Ajay       Ash             50000        500          5

SQL>
```

Adding Calculator:



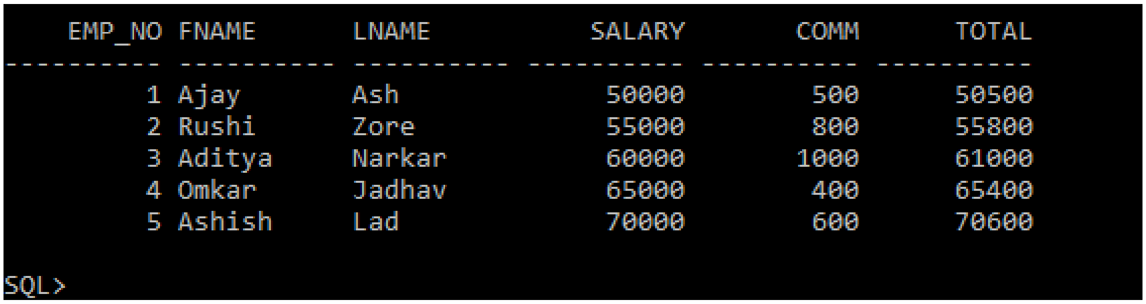
Calculator

Step name: Calculator

Fields:

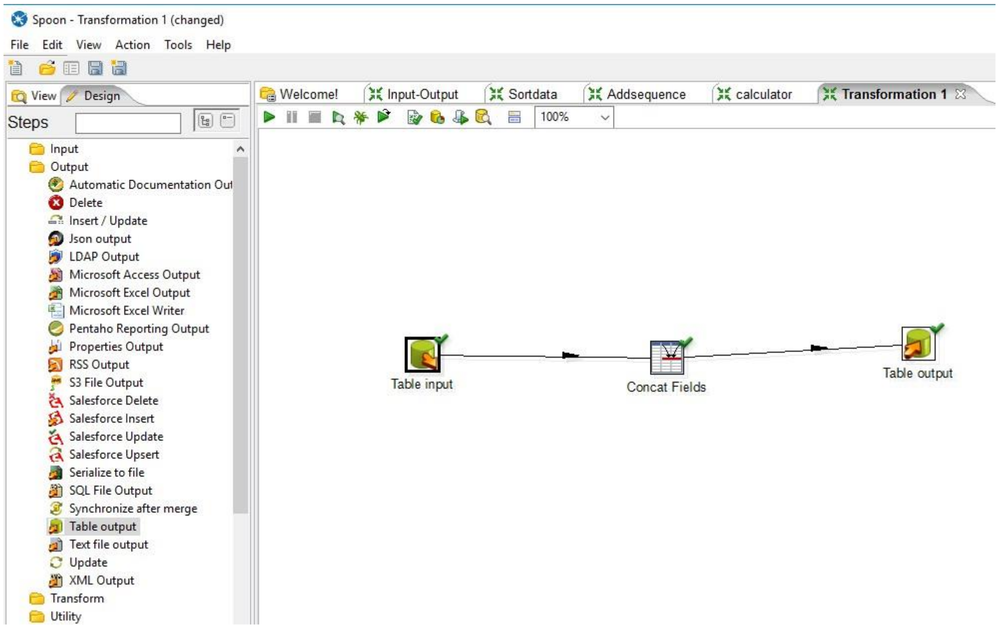
#	New field	Calculation	Field A	Field B	Field C	Value type	Length	Precision	Remove
1	Total	A + B	SALARY	COMM		Number	5		N

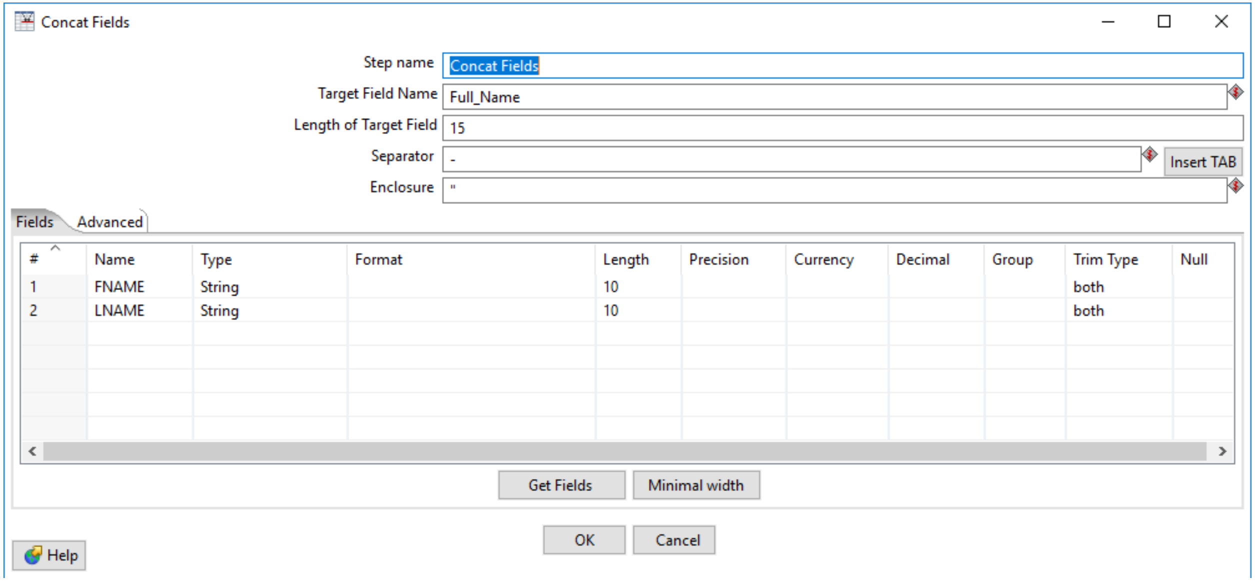
SQL> Select * from emp26;



```
    EMP_NO FNAME      LNAME          SALARY       COMM      TOTAL
---------- ---------- ---------- ---------- ---------- ----------
         1 Ajay       Ash             50000        500      50500
         2 Rushi      Zore            55000        800      55800
         3 Aditya     Narkar          60000       1000      61000
         4 Omkar      Jadhav          65000        400      65400
         5 Ashish     Lad             70000        600      70600

SQL>
```

Concatenation of two fields:





Step name: Concat Fields

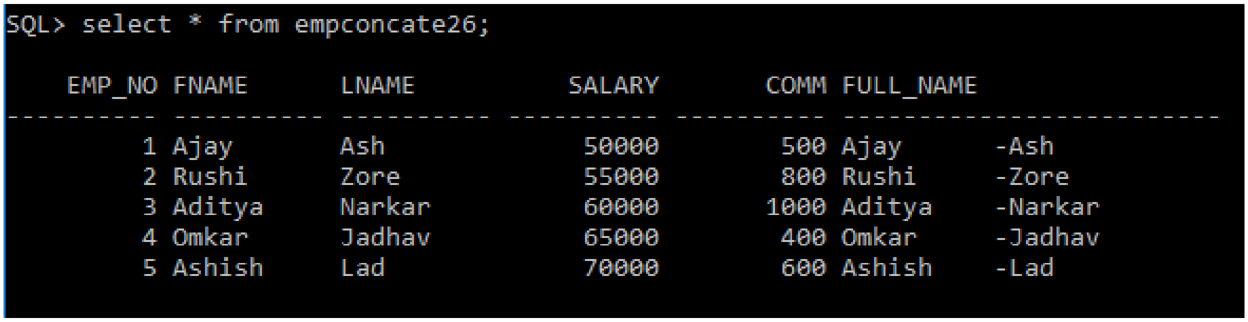
Target Field Name: Full_Name

Length of Target Field: 15

Separator: -

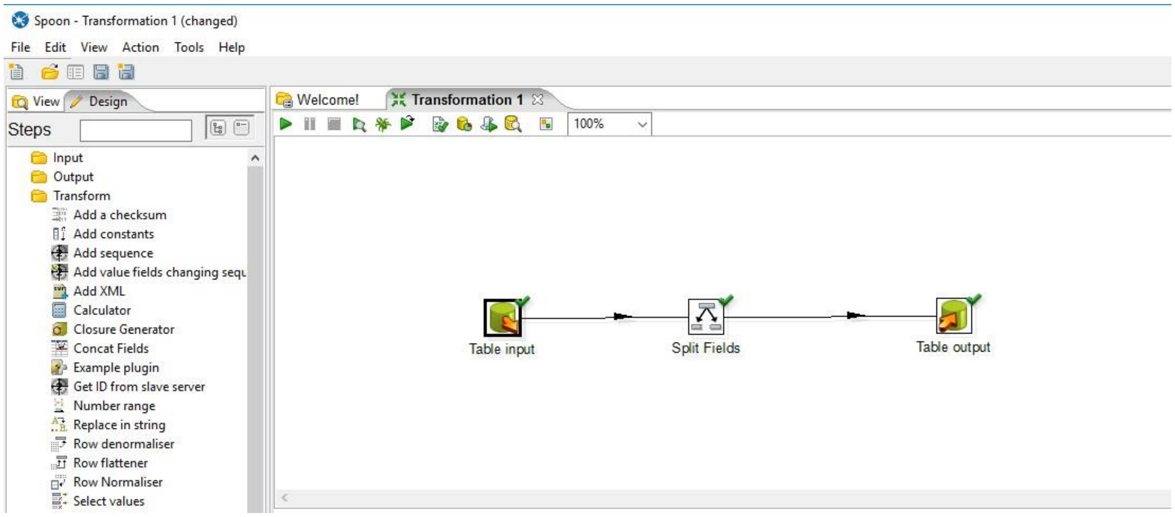
Enclosure: "

#	Name	Type	Format	Length	Precision	Currency	Decimal	Group	Trim Type	Null
1	FNAME	String		10					both	
2	LNAME	String		10					both	

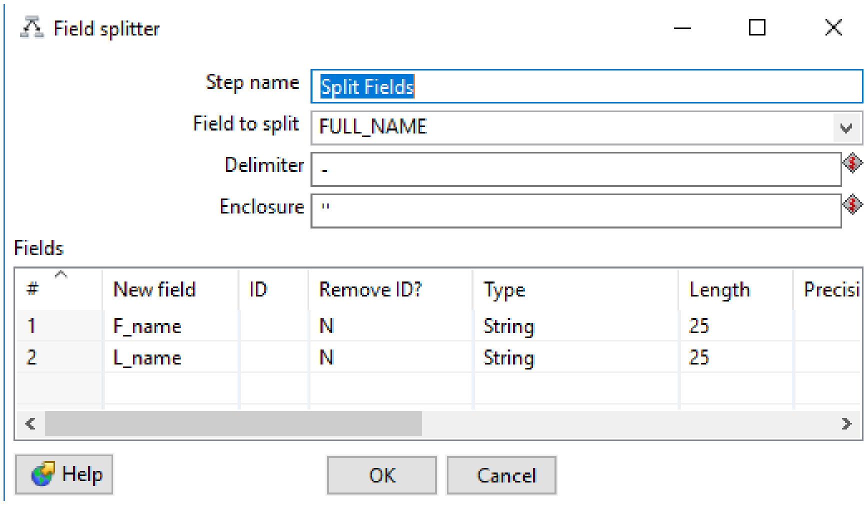


```
SQL> select * from empconcate26;

    EMP_NO FNAME      LNAME          SALARY       COMM FULL_NAME
---------- ---------- ---------- ---------- ---------- -------------------------
         1 Ajay       Ash             50000        500 Ajay      -Ash
         2 Rushi      Zore            55000        800 Rushi     -Zore
         3 Aditya     Narkar          60000       1000 Aditya    -Narkar
         4 Omkar      Jadhav          65000        400 Omkar     -Jadhav
         5 Ashish     Lad             70000        600 Ashish    -Lad
```

Splitting of two fields:



For Splitting Any Field we have to take concate table:



Step name: Split Fields

Field to split: FULL_NAME

Delimiter: -

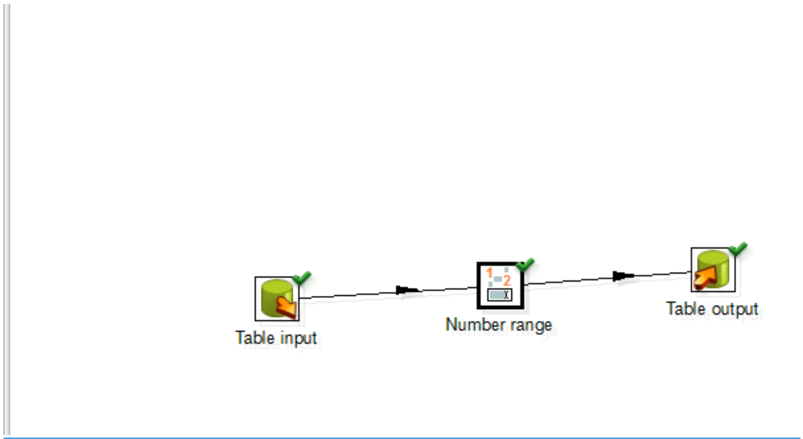
Enclosure: "

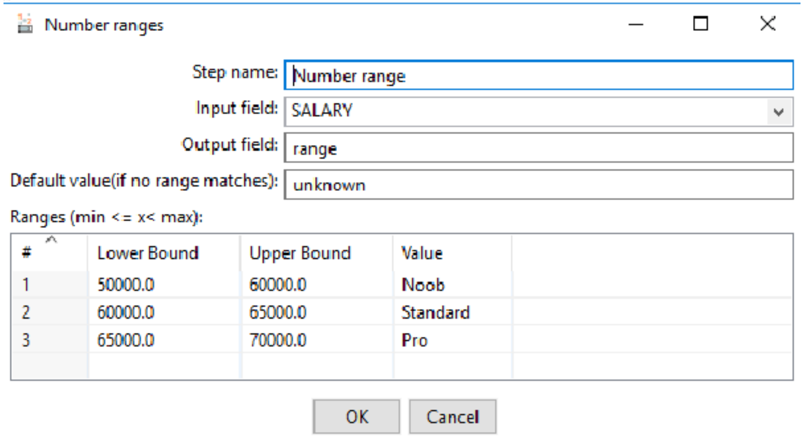
Fields

#	New field	ID	Remove ID?	Type	Length	Precisi
1	F_name		N	String	25	
2	L_name		N	String	25	

```
SQL> Select * from split78;

EMP_NO  FNAME   LNAME   SALARY COMM   F_NAME  L_NAME

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

   1        Ajay     Ash     50000   500      Ajay     Ash

   2        Rushi    Wani    55000   800      Rushi    Wani

   3        Aditya   Narkar  60000   1000     Aditya   Narkar

   4        Omkar    Jadha   65000   400      Omkar    Jadha
                     v                                 v
   5        Ashish   Lad     70000   600      Ashish   Lad
```

Number Range:




Number ranges

Step name: Number range

Input field: SALARY

Output field: range

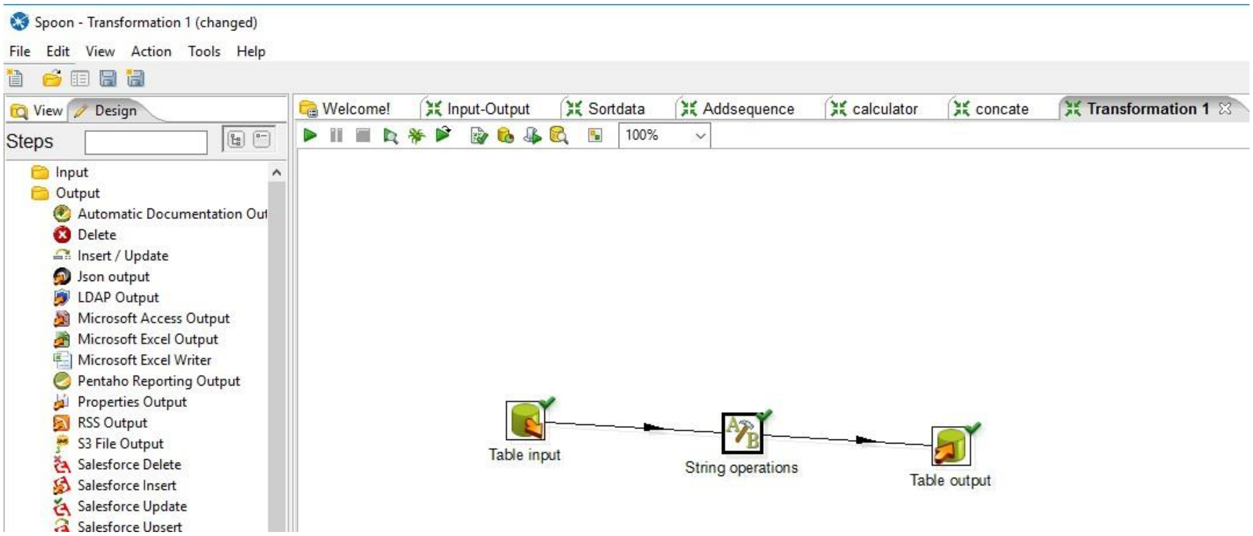
Default value(if no range matches): unknown

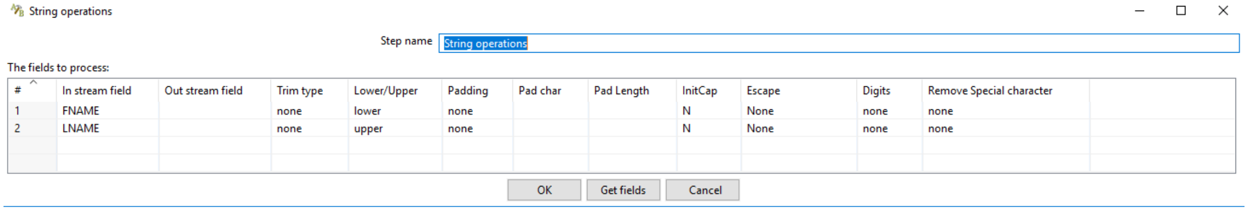
Ranges (min <= x< max):

#	Lower Bound	Upper Bound	Value
1	50000.0	60000.0	Noob
2	60000.0	65000.0	Standard
3	65000.0	70000.0	Pro

```
SQL> Select * from empnorange26;
EMP_NO  FNAME   LNAME   SALARY COMM    RANGE
------------------ -------------- ------------- -------------- ------------- ---------------
  1      Ajay    Ash     50000   500    Noob
  2      Rushi   Wani    55000   800    Noob
  3      Aditya  Narkar  60000   1000   Standard
  4      Omkar   Jadhav  65000   400    Pro
  5      Ashish  Lad     70000   600    Unknow
                                        n
```

String Operations:

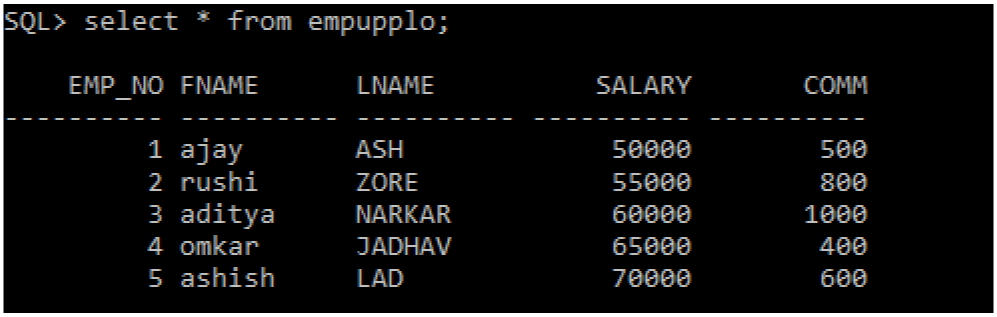




Step name: String operations

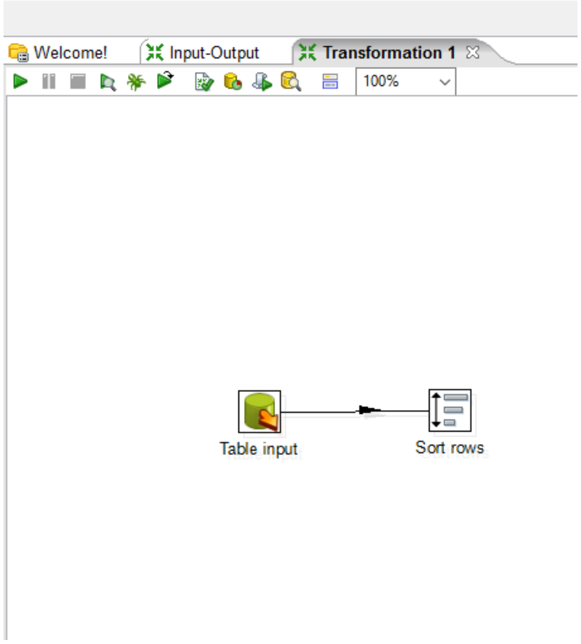
The fields to process:

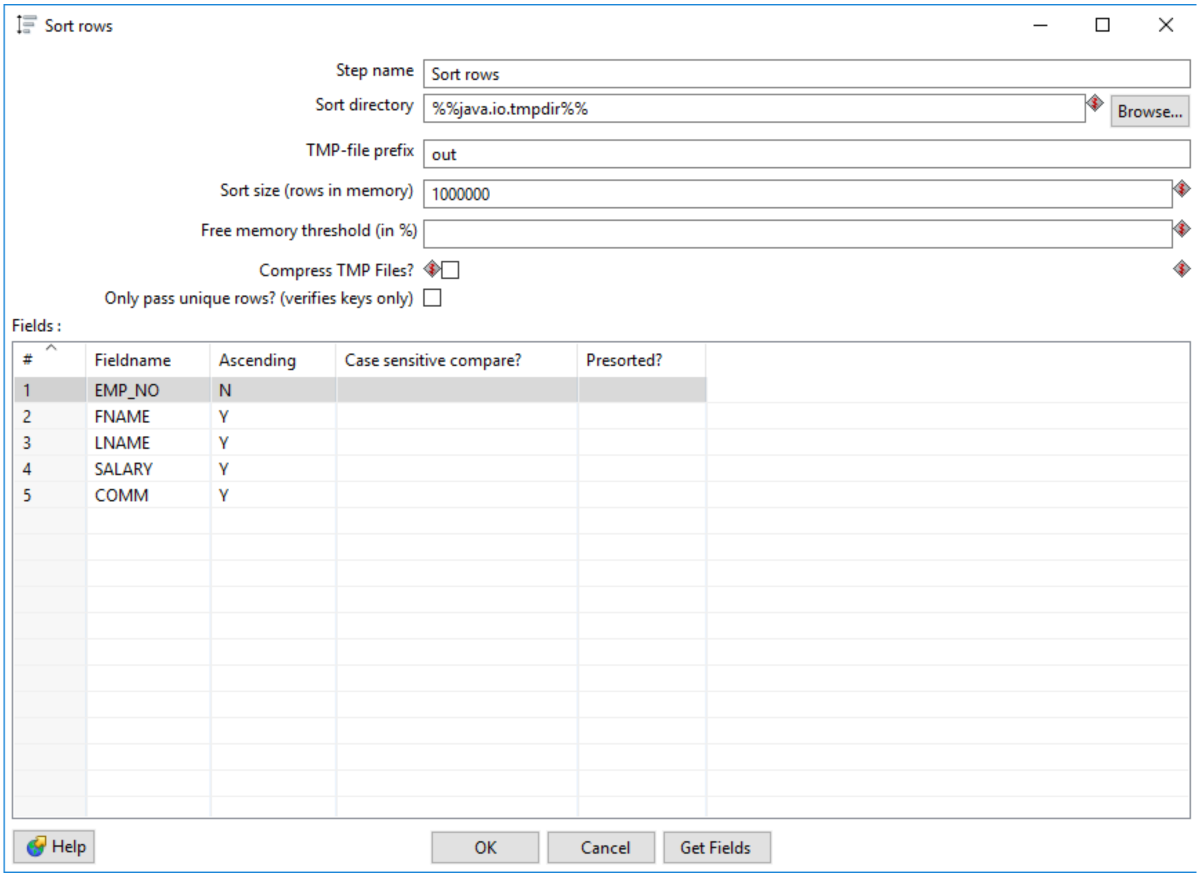
#	In stream field	Out stream field	Trim type	Lower/Upper	Padding	Pad char	Pad Length	InitCap	Escape	Digits	Remove Special character
1	FNAME		none	lower	none			N	None	none	none
2	LNAME		none	upper	none			N	None	none	none



```
SQL> select * from empupplo;

    EMP_NO FNAME      LNAME          SALARY       COMM
---------- ---------- ---------- ---------- ----------
         1 ajay       ASH             50000        500
         2 rushi      ZORE            55000        800
         3 aditya     NARKAR          60000       1000
         4 omkar      JADHAV          65000        400
         5 ashish     LAD             70000        600
```

Sorting data:





Step name: Sort rows

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

TMP-file prefix: out

Sort size (rows in memory): 1000000

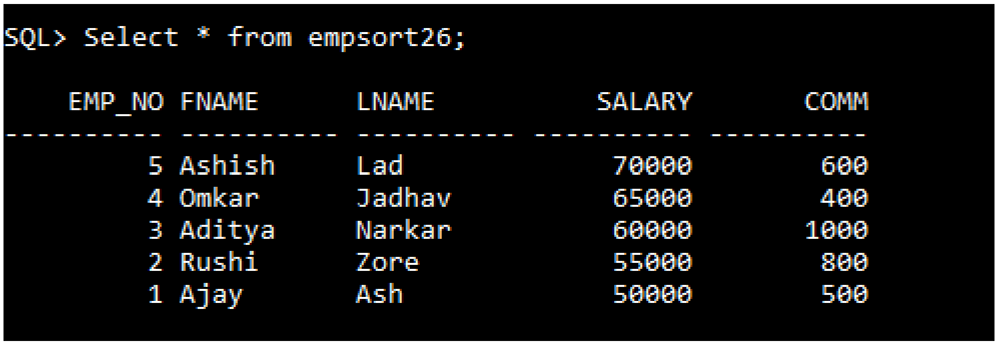
Free memory threshold (in %):

Compress TMP Files? □

Only pass unique rows? (verifies keys only) □

Fields :

#	Fieldname	Ascending	Case sensitive compare?	Presorted?
1	EMP_NO	N		
2	FNAME	Y		
3	LNAME	Y		
4	SALARY	Y		
5	COMM	Y		



```
SQL> Select * from empsort26;

    EMP_NO FNAME      LNAME          SALARY       COMM
---------- ---------- ---------- ---------- ----------
         5 Ashish     Lad             70000        600
         4 Omkar      Jadhav          65000        400
         3 Aditya     Narkar          60000       1000
         2 Rushi      Zore            55000        800
         1 Ajay       Ash             50000        500
```

6. Introduction to R programming and Data acquisition

Install packages , Loading packages

Data types, checking type of variable, printing variable and objects (Vector, Matrix, List, Factor, Data frame, Table) cbind-ing and rbind-ing

- **Reading and Writing data.**
  setwd(), getwd(), data(), rm(),
- **Attaching and Detaching data.**

Reading data from the consol.

Loading data from different data sources.(CSV, Excel).

Source code:

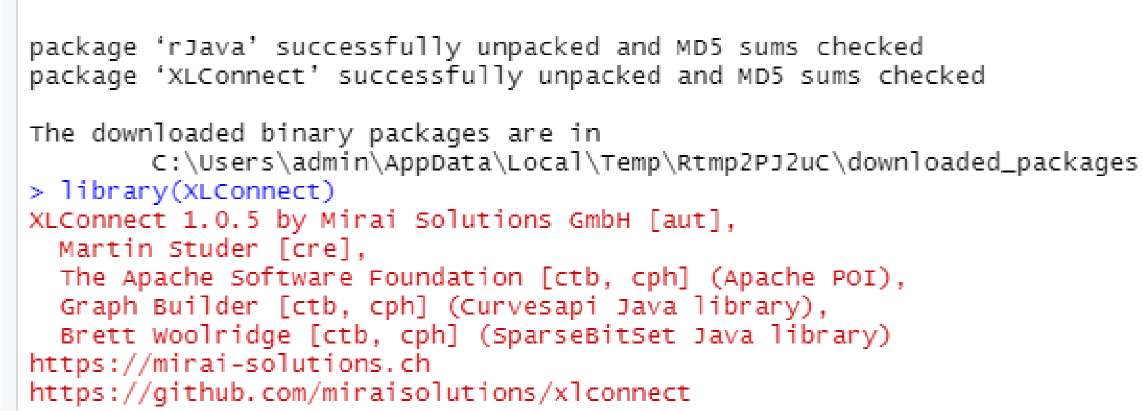
```
> setwd("C:/Users/admin/Desktop/R-prog")

> getwd()

[1] "C:/Users/admin/Desktop/R-prog"

> dir()

character(

0)

> x<-2

> x

[1] 2

> print(x)

[1] 2

> is.character(x)

[1] FALSE

> is.integer(x)

[1] FALSE

> is.numeric(x)

[1] TRUE
```

```
> y<-"5.25"
> as.integer(y)
[1] 5
> x<-c(11.3,27.5,33.8)
> y<-vector("logical", length=10)
> length(x)
[1] 3
> length(y)
[1] 10
> y<-c(4,5,6,10)
> 5*x
[1] 56.5 137.5 169.0
> x*y
[1] 45.2 137.5 202.8 113.0
 a<-c(2,4,6)
> b<-c(1,3,5)
> a*b
[1] 2 12 30
> a^b
[1]    2   64 7776
> m<-matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3)
> dim(m)
[1] 3 3
> m
     [,1] [,2] [,3]
[1,]   11   55   66
[2,]   12   60   72
[3,]   13   65   78
> attributes(m)
```

```
$dim
[1] 3 3

> m<-matrix(c(11,12,13,55,60,65,66,72,78),nrow=3,ncol=3,byrow = TRUE)
> x<-c(1,2,3)
> y<-c(11,12,13)
> cbind(x,y)
     x y
[1,] 1 11
[2,] 2 12
[3,] 3 13
> rbind(x,y)
  [,1] [,2] [,3]
x    1    2    3
y   11   12   13
> n<-matrix(c(4,5,6,14,15,16,24,25,26),nrow=3,ncol=3)
> q<-m+n
> q
     [,1] [,2] [,3]
[1,]   15   26   37
[2,]   60   75   90
[3,]   72   88  104
> o<-matrix(c(4,5,6,14,15,16),nrow=3,ncol=2)
> o
     [,1] [,2]
[1,]    4   14
[2,]    5   15
[3,]    6   16
> r<-m %*% o
> r
     [,1] [,2]
[1,] 182  542
[2,] 910 2710
[3,] 1092 3252
> mdash<-t(m)
> mdash
     [,1] [,2] [,3]
[1,]   11   55   66
[2,]   12   60   72
[3,]   13   65   78
> s<-matrix(c(4,5,6,14,15,16,24,25,26), nrow=3,ncol=3,byrow=TRUE)
> s
     [,1] [,2] [,3]
[1,]    4    5    6
[2,]   14   15   16
[3,]   24   25   26
> dt<-det(s)
> dt
[1] 1.110223e-14
> x<-list(1,"p",TRUE,3+4i)
> x
[[1]]
[1] 1

[[2]]
[1] "p"

[[3]]
```

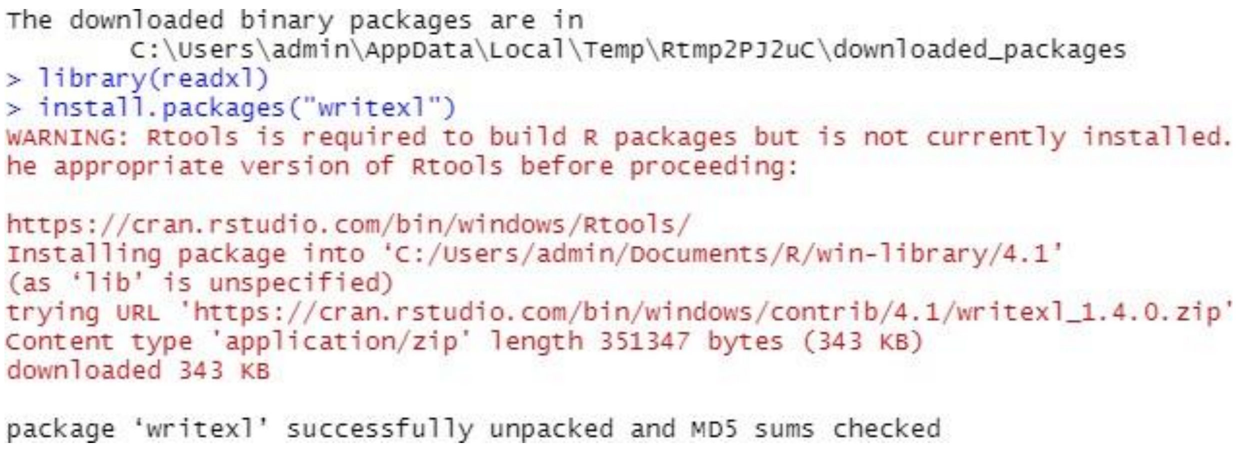
```
[1] TRUE


[[4]]
[1] 3+4i


> student_names<-c("Vinit","Ajay","Aditya")
> position<-c("First","Second","Third")
 student_id<-c(1,2,3)
> data<-data.frame(student_id,student_names,position)
> data
  student_id student_names position
1          1    Vinit              First
2          2          Ajay   Second
3          3        Aditya    Third
> data$student_id
[1] 1 2 3
> data$student_names
[1] "Vinit" "Ajay"                "Aditya"
> nrow(data)
[1] 3
> ncol(data)
[1] 3
> names(data)
[1] "student_id"    "student_names" "position"
> 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
> install.packages("XLConnect")
```



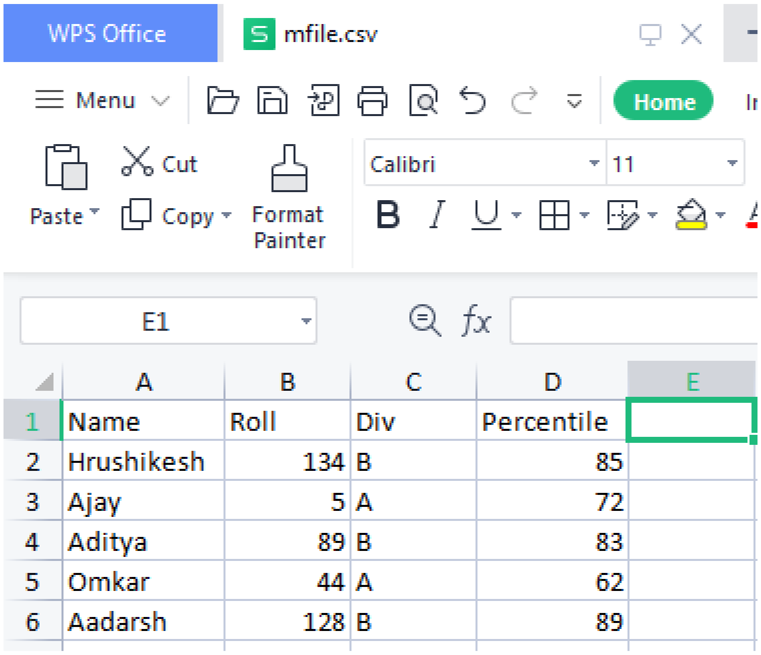
```
package ‘rJava’ successfully unpacked and MD5 sums checked
package ‘XLConnect’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\Rtmp2PJ2uC\downloaded_packages
> library(XLConnect)
XLConnect 1.0.5 by Mirai Solutions GmbH [aut],
  Martin Studer [cre],
  The Apache Software Foundation [ctb, cph] (Apache POI),
  Graph Builder [ctb, cph] (Curvesapi Java library),
  Brett Woolridge [ctb, cph] (SparseBitSet Java library)
https://mirai-solutions.ch
https://github.com/miraisolutions/xlconnect
```

```
>install.packages("readxl")
```



```
The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\Rtmp2PJ2uC\downloaded_packages
> library(readxl)
> install.packages("writexl")
WARNING: Rtools is required to build R packages but is not currently installed.
he appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/admin/Documents/R/win-library/4.1’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/writexl_1.4.0.zip'
Content type 'application/zip' length 351347 bytes (343 KB)
downloaded 343 KB

package ‘writexl’ successfully unpacked and MD5 sums checked
```

```
>library(writexl)
```



	A	B	C	D	E
1	Name	Roll	Div	Percentile	
2	Hrushikesh	134	B	85	
3	Ajay	5	A	72	
4	Aditya	89	B	83	
5	Omkar	44	A	62	
6	Aadarsh	128	B	89	

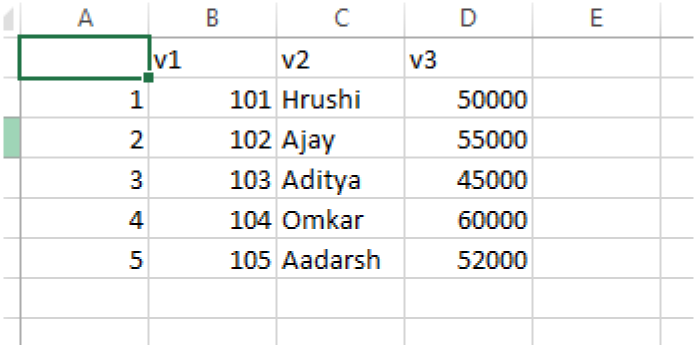
```
>dataT <- read.table("mfile.csv", sep =",", header = T)
>dataT
     Name Roll Div
Percentile 1 Vinit  134   B
                              85
2     Ajay    5   A     72
3   Aditya   89   B     83
4    Omkar   44   A     62
5   Aadarsh  128   B    89
>dim(dataT
) [1] 5 4
>head(dataT, 2)
     Name Roll Div
Percentile 1 Vinit  134   B
                              85
2     Ajay    5   A     72
>tail(dataT, 2)
   Name Roll Div Percentile
4  Omkar   44   A     62
```

```
5 Aadarsh  128   B    89
> dataX <- XLConnect:: readWorksheetFromFile("mfile.xlsx",sheet=1)

> dataX

     Name Roll Div

Percentile 1 Vinit  134   B

                              85

2     Ajay    5   A     72

3    Aditya   89   B     83

4     Omkar   44   A     62

5   Aadarsh  128   B    89
```

7. Implementation of Data preprocessing techniques like, Naming and Renaming variables, adding a new variable. Dealing with missing data. Dealing with categorical data. Data reduction using subsetting

Source code:

```
setwd("C:/Users/admin/Desktop/R-prog")

> install.packages(“dplyr”)

Error: unexpected input in "install.packages(“"

> data2 = read.table(file="naming_var_table.csv", sep = ",")

> data2

  V1  V2      V3    V4

1 NA  v1      v2    v3

 2 1 101  Hrushi 50000

 3 2 102    Ajay 55000

 4 3 103  Aditya 45000

 5 4 104   Omkar 60000

 6 5 105 Aadarsh 52000

> data2 = read.csv(file="naming_var_table.csv", col.names=c("Sno",
"NAME","SALARY"))

Warning message:

In read.table(file = file, header = header, sep = sep, quote = quote,  :

  header and 'col.names' are of different lengths
```



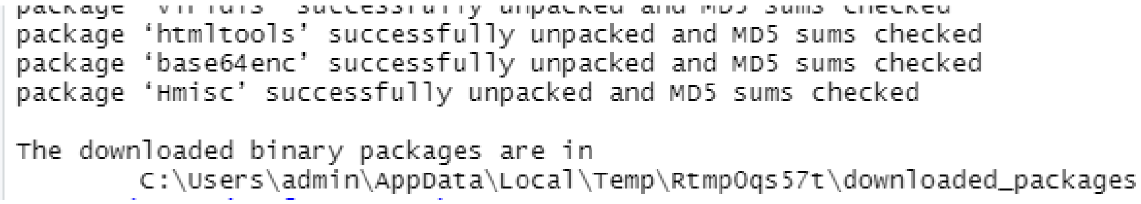
A	B	C	D	E
	v1	v2	v3	
1	101	Hrushi	50000	
2	102	Ajay	55000	
3	103	Aditya	45000	
4	104	Omkar	60000	
5	105	Aadarsh	52000	

```
> data2 = read.csv(file="naming_var_table.csv", col.names=c("Rno","Sno",
"NAME","SALARY"))

> data2

  Rno Sno    NAME SALARY

1   1 101 Hrushi  50000

2   2 102    Ajay  55000

3   3 103 Aditya  45000

4   4 104   Omkar  60000

5   5 105 Aadarsh  52000

> NA + 4

[1] NA

> V <- c(1,2,NA,3)

> median(v)

Error in median(v) : object 'v' not found

> median(V)

[1] NA

> median(V, na.rm = T)

[1] 2

> is.na(V)

[1] FALSE FALSE TRUE FALSE

> naVals <- is.na(V)

> V[!naVals]

[1] 1 2 3

> V[complete.cases(V)]

[1] 1 2 3

> dataC <- read.csv(file ="naming_var_table.csv")

> > dataC

Error: unexpected '>' in ">"

> dataC <- read.csv(file ="naming_var_table.csv")
```

```
>
> dataC
  X  v1      v2    v3
1 1 101  Hrushi 50000
2 2 102    Ajay 55000
3 3 103  Aditya 45000
4 4 104   Omkar 60000
5 5 105 Aadarsh 52000
> dataCompleteCases <- dataC[complete.cases(dataC),]
> dataCompleteCases
  X  v1      v2    v3
1 1 101  Hrushi 50000
2 2 102    Ajay 55000
3 3 103  Aditya 45000
4 4 104   Omkar 60000
5 5 105 Aadarsh 52000
```

IMPUTATION:

```
> install.packages("Hmisc")
```



```
package 'viridis' successfully unpacked and MD5 sums checked
package 'htmltools' successfully unpacked and MD5 sums checked
package 'base64enc' successfully unpacked and MD5 sums checked
package 'Hmisc' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\RtmpOqs57t\downloaded_packages
```

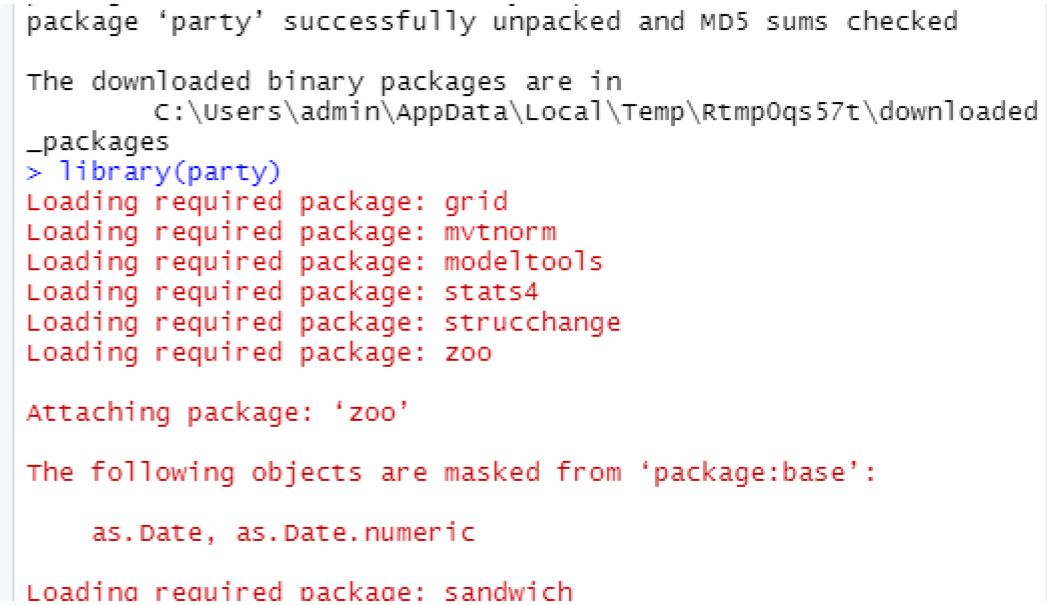
```
> library(Hmisc)
Loading required package: lattice
Loading required package: survival
Loading required package: Formula
Loading required package: ggplot2
```

```
Attaching package: ‘Hmisc’

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

    format.pval, units

> x <- impute(x, fun = mean)
> x
   1    2    3    4    5    6    7 
 1.0 2.0 3.0 2.8* 4.0 4.0 2.8* 
> x <- impute(x, fun = median)
> x
   1    2    3    4    5    6    7 
 1.0 2.0 3.0 2.8* 4.0 4.0 2.8* 

> x = c(1,2,3,NA,4,4,NA)
> gender_vector <- c("Male", "Female", "Female", "Male", "Male")
> class(gender_vector)
[1] "character"
> 
> factor_gender_vector <-factor(gender_vector)
> class(factor_gender_vector)
[1] "factor"
> age <- c(40, 49, 48, 40, 67, 52, 53)
> salary <- c(103200, 106200, 150200, 10606, 10390, 14070, 10220)
> gender <- c("male", "male", "transgender",
+             "female", "male", "female", "transgender")
> employee<- data.frame(age, salary, gender)
> employee
  age salary      gender
1  40 103200        male
2  49 106200        male
3 48 150200 transgender
4  40  10606      female
5  67  10390        male
6  52  14070      female
7 53 10220 transgender
> wfact = cut(employee$age, 3, labels=c('Young', 'Medium', 'Aged'))
> table(wfact)
wfact
 Young Medium   Aged
     4      2      1
> wfact = cut(employee$salary, 2, labels=c('Below Lakh', 'Above Lakh'))
> table(wfact)
wfact
Below Lakh Above Lakh
         4          3
```

8.Implementation and analysis of Classification algorithms like Naive Bayesian, K-Nearest Neighbor, ID3 , C4.5

Decision Tree

> install.packages("party")



```
package ‘party’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\RtmpOqs57t\downloaded
_packages
> library(party)
Loading required package: grid
Loading required package: mvtnorm
Loading required package: modeltools
Loading required package: stats4
Loading required package: strucchange
Loading required package: zoo

Attaching package: ‘zoo’

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

    as.Date, as.Date.numeric

Loading required package: sandwich
```

>print(head(readingSkills))

nativeSpeaker age shoeSize

score 1 yes 5 24.83189

32.29385

2 yes 6 25.95238 36.63105

3 no 11 30.42170 49.60593

4 yes 7 28.66450 40.28456

5 yes 11 31.88207 55.46085

6 yes 10 30.07843 52.83124

>input.dat <- readingSkills[c(1:105),

+

+]

>png(file = "decision_tree.png")

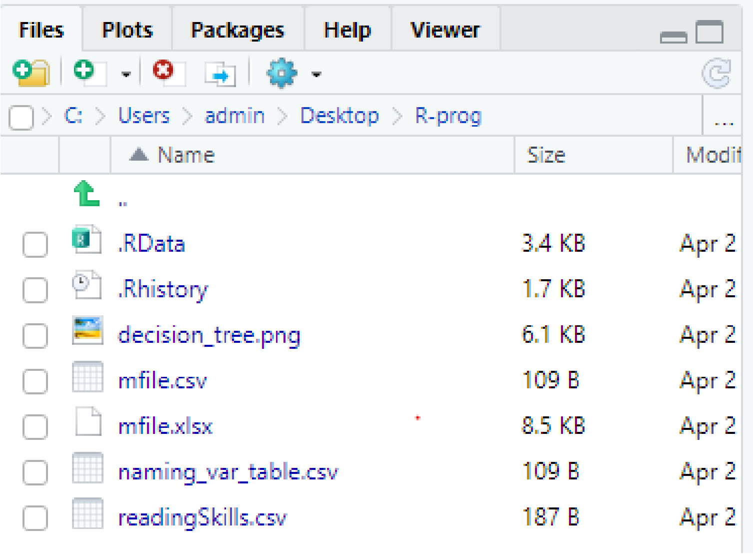
>output.tree <- ctree(

+ nativeSpeaker ~ age + shoeSize + score,

+ data = input.dat)

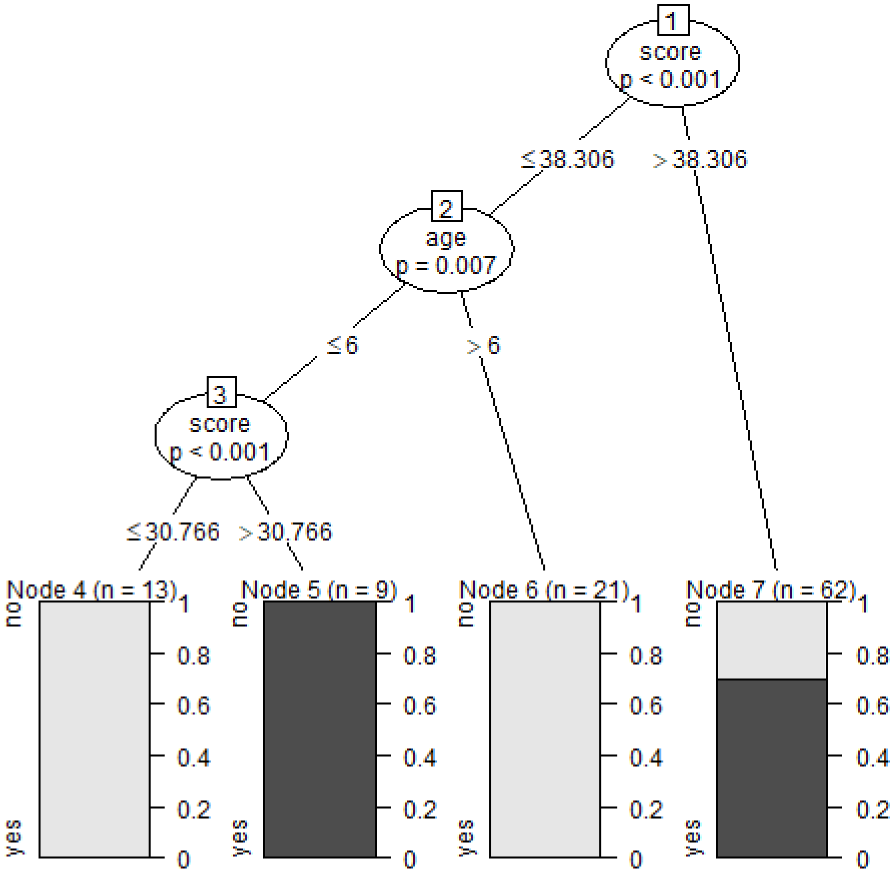
```
> plot(output.tree)
> dev.off(
) null
device
      1
```

Output:



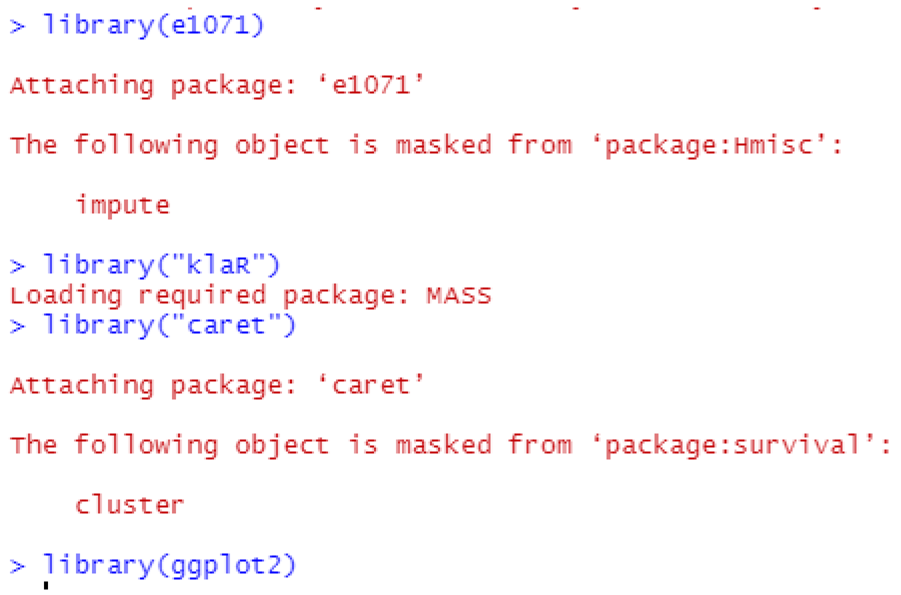
C: > Users > admin > Desktop > R-prog

	▲ Name	Size	Modif
	..		
□	.RData	3.4 KB	Apr 2
□	.Rhistory	1.7 KB	Apr 2
□	decision_tree.png	6.1 KB	Apr 2
□	mfile.csv	109 B	Apr 2
□	mfile.xlsx	8.5 KB	Apr 2
□	naming_var_table.csv	109 B	Apr 2
□	readingSkills.csv	187 B	Apr 2

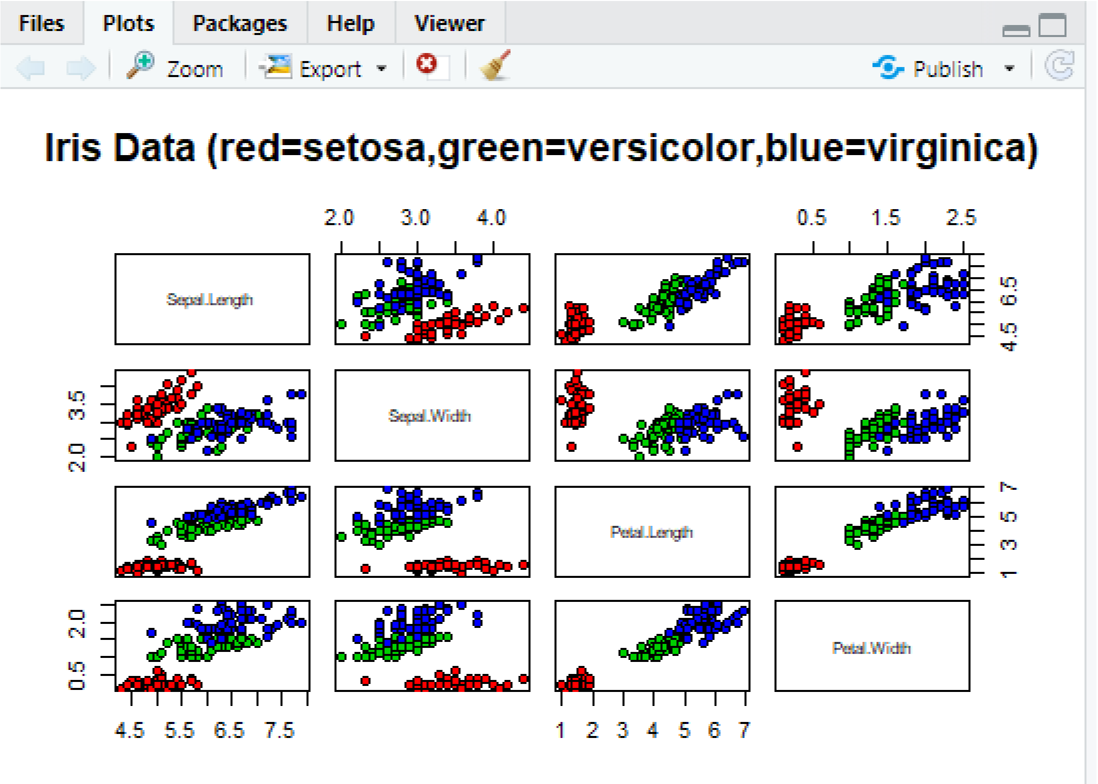

1
score
p < 0.001
≤ 38.306
> 38.306
2
age
p = 0.007
≤ 6
> 6
3
score
p < 0.001
≤ 30.766
> 30.766
Node 4 (n = 13)
Node 5 (n = 9)
Node 6 (n = 21)
Node 7 (n = 62)
no
yes
1
0.8
0.6
0.4
0.2
0

Classification: Naive Bayes

```
>install.packages("e1071")

>install.packages("klaR")

>install.packages("caret")
```



```
> library(e1071)

Attaching package: ‘e1071’

The following object is masked from ‘package:Hmisc’:

    impute

> library("klaR")
Loading required package: MASS
> library("caret")

Attaching package: ‘caret’

The following object is masked from ‘package:survival’:

    cluster

> library(ggplot2)
> data(iris)

> head(iris)

  Sepal.Length Sepal.Width Petal.Length Petal.Width

1       5.1       3.5       1.4       0.2

2       4.9       3.0       1.4       0.2

3       4.7       3.2       1.3       0.2

4       4.6       3.1       1.5       0.2

5       5.0       3.6       1.4       0.2

6       5.4       3.9       1.7       0.4

  Specie

s 1

setosa

2 setosa

3 setosa

4 setosa

5 setosa

6 setosa

> unique(iris$Species)

[1] setosa   versicolor virginica

Levels: setosa versicolor

virginica

>     pairs(iris[1:4],     main="Iris     Data     (red=setosa,green=versicolor,blue=virginica)",
      pch=21, bg=c("red","green3","blue")[unclass(iris$Species)])
```



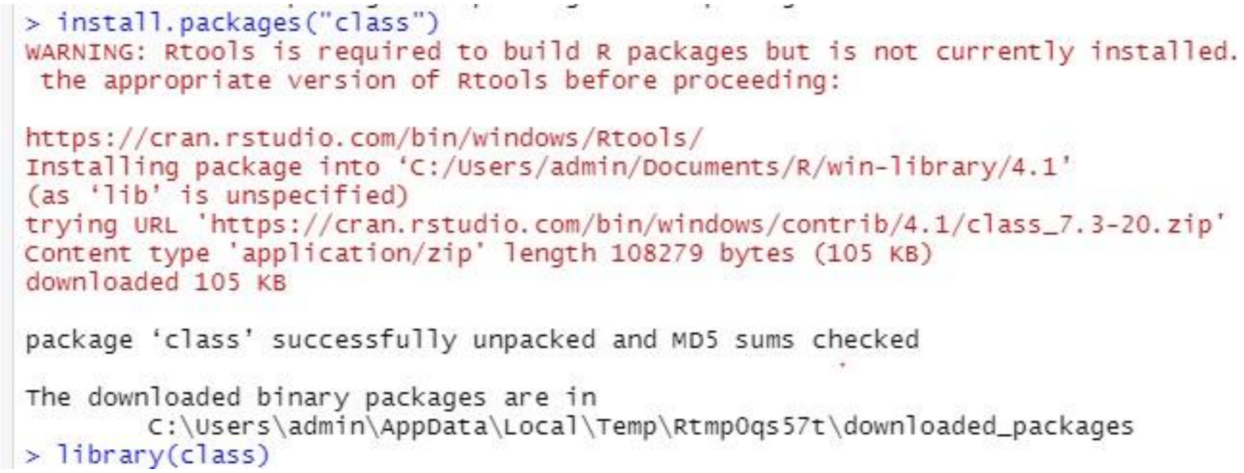
```
>index = sample(nrow(iris), floor(nrow(iris) * 0.7)) #70/30 split.

>train = iris[index,] test = iris[-index,]

Error: unexpected symbol in "train = iris[index,] test"

>index = sample(nrow(iris), floor(nrow(iris) * 0.7))

>train = iris[index,]

>test = iris[-index,]

>xTrain = train[,-5]

>yTrain = train$Species

>xTest = test[,-5]

>yTest = test$Species

>model = train(xTrain,yTrain,'nb',trControl=trainControl(method='cv',number=10))

>model

Naive

Bayes


105 samples
```

```
 4 predictor
 3 classes: 'setosa', 'versicolor', 'virginica'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 95, 95, 94, 94, 93,
95, ... Resampling results across tuning
parameters:

 usekernel Accuracy Kappa
 FALSE  0.9500000
 0.9244616
  TRUE   0.9409091  0.9105376

Tuning parameter 'fL' was

Tuning parameter 'adjust'
was held constant at a
value of 1 Accuracy was
used to select the optimal
model using the largest
value.
The final values used for the
 model were fL = 0,
 usekernel = FALSE and
 adjust = 1.
> prop.table(table(predict(model$finalModel,xTest)$class,yT
        est)) yTest
          setosa versicolor
 setosa   0.3111111
 0.0000000
```

```
versicolor 0.0000000 0.4000000
virginica 0.0000000
        0.0000000 yTest
```

```
          virginica
setosa
               0.000000
0
versicolor 0.0000000
virginica 0.2888889
```

K nearest Neighbour



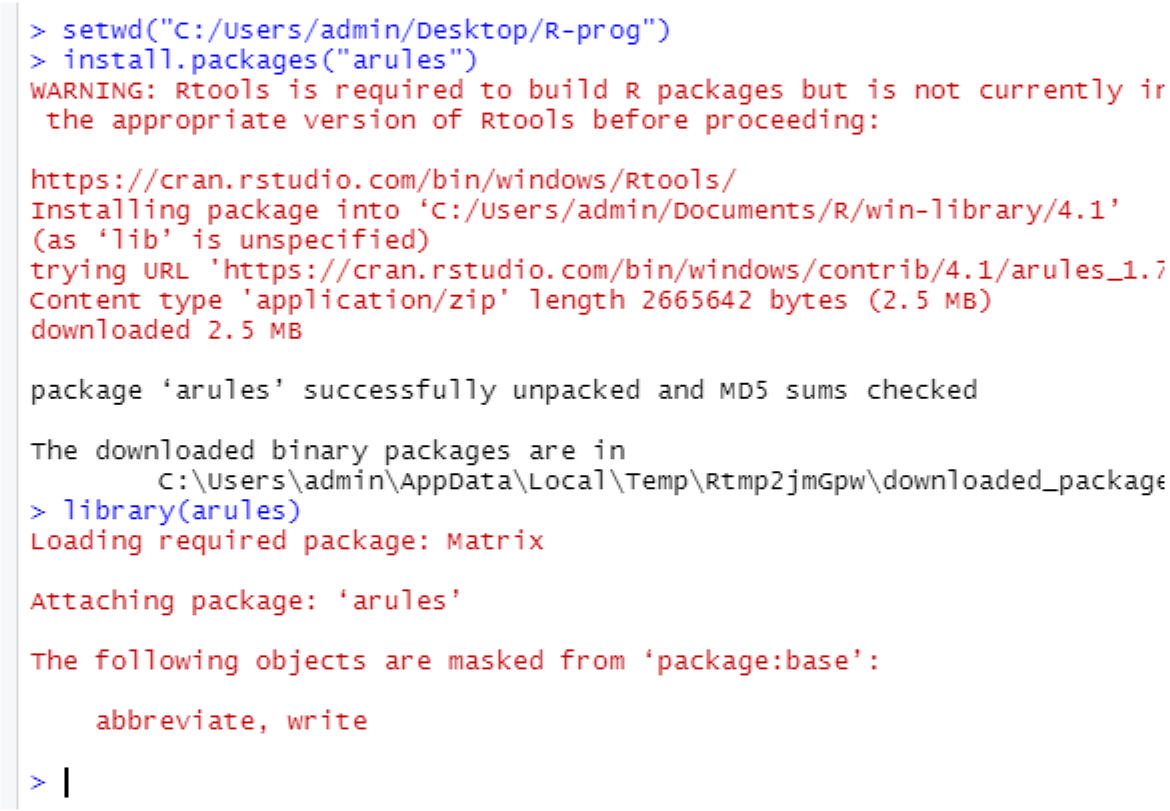
```
> install.packages("class")
WARNING: Rtools is required to build R packages but is not currently installed.
 the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into 'C:/Users/admin/Documents/R/win-library/4.1'
(as 'lib' is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/class_7.3-20.zip'
Content type 'application/zip' length 108279 bytes (105 KB)
downloaded 105 KB

package 'class' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\Rtmp0qs57t\downloaded_packages
> library(class)
```

```
> df <- data(iris)

> head(iris)

  Sepal.Length Sepal.Width

1       5.1       3.5

2       4.9       3.0

3       4.7       3.2

4       4.6       3.1

5       5.0       3.6

6       5.4       3.9

  Petal.Length Petal.Width   Species

1       1.4       0.2  setosa

2       1.4       0.2  setosa

3       1.3       0.2  setosa

4       1.5       0.2  setosa

5       1.4       0.2  setosa

6       1.7       0.4  setosa

> ran <- sample(1:nrow(iris), 0.9 *  nrow(iris))

> nor <-function(x) { (x -min(x))/(max(x)-min(x))  }

>
```

```
> iris_norm <- as.data.frame(lapply(iris[,c(1,2,3,4)], nor))
> summary(iris_norm)
 Sepal.Length  Sepal.Width
 Min.  :0.0000  Min.  :0.0000
 1st Qu.:0.2222 1st
 Qu.:0.3333
 Median :0.4167 Median :0.4167
 Mean  :0.4287  Mean  :0.4406
 3rd         Qu.:0.5833         3rd
 Qu.:0.5417     Max.     :1.0000
 Max.  :1.0000
  Petal.Length  Petal.Width
 Min.  :0.0000  Min.  :0.00000
 1st Qu.:0.1017 1st
 Qu.:0.08333
 Median :0.5678 Median :0.50000
 Mean  :0.4675  Mean  :0.45806
 3rd          Qu.:0.6949          3rd
 Qu.:0.70833     Max.     :1.0000
 Max.  :1.00000
> iris_train <- iris_norm[ran,]
> iris_test <- iris_norm[-ran,]
> iris_target_category <- iris[ran,5]
> iris_test_category <- iris[-ran,5]> library(class)
> pr <- knn(iris_train,iris_test,cl=iris_target_category,k=13)
> tab <- table(pr,iris_test_category)
> accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
> accuracy(ta
b) [1] 100
```

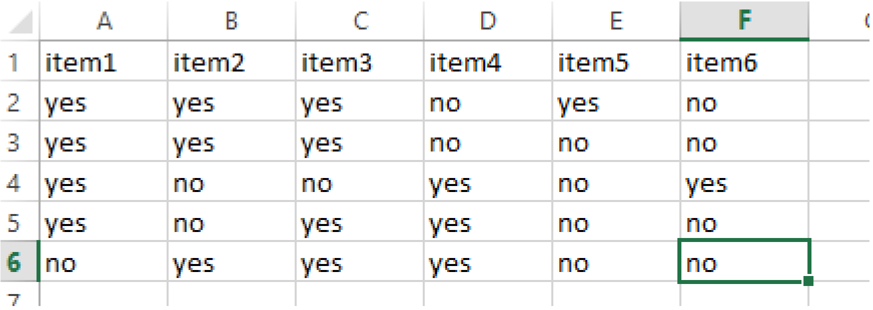
9. Implementation and analysis of Apriori Algorithm using Market Basket Analysis.

Source code:



```
> setwd("C:/Users/admin/Desktop/R-prog")
> install.packages("arules")
WARNING: Rtools is required to build R packages but is not currently ir
 the appropriate version of Rtools before proceeding:

https://cran.rstudio.com/bin/windows/Rtools/
Installing package into ‘C:/Users/admin/Documents/R/win-library/4.1’
(as ‘lib’ is unspecified)
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/arules_1.7
Content type 'application/zip' length 2665642 bytes (2.5 MB)
downloaded 2.5 MB

package ‘arules’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\admin\AppData\Local\Temp\Rtmp2jmGpw\downloaded_package
> library(arules)
Loading required package: Matrix

Attaching package: ‘arules’

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

    abbreviate, write

> |
```

Shopping_basket.csv:



	A	B	C	D	E	F
1	item1	item2	item3	item4	item5	item6
2	yes	yes	yes	no	yes	no
3	yes	yes	yes	no	no	no
4	yes	no	no	yes	no	yes
5	yes	no	yes	yes	no	no
6	no	yes	yes	yes	no	no

>length(shopping)

[1] 6

```
> dim(shopping)

[1] 5 6

> rules<- apriori(shopping)

Apriori

Parameter specification:

 confidence minval smax arem aval originalSupport maxtime support minlen
maxlen target ext

        0.8   0.1   1 none FALSE      TRUE    5    0.1     1    10 rules TRUE


Algorithmic control:

 filter tree heap memopt load sort verbose

    0.1 TRUE TRUE  FALSE TRUE   2        TRUE


Absolute minimum support count: 0


set item appearances ...[0 item(s)] done [0.00s].

set transactions ...[14 item(s), 5 transaction(s)] done

[0.00s]. sorting and recoding items ... [14 item(s)] done

[0.00s]. creating transaction tree ... done [0.00s].

checking subsets of size 1 2 3 4 5 6 done

[0.00s]. creating S4 object ... done [0.00s].

Warning message:

Column(s) 1, 2, 3, 4, 5, 6 not logical or factor. Applying default discretization (see '?
discretizeDF').


> inspect(rules)

      lhs                  rhs       support confidence coverage lift

                          count [1]   {}             => {item1= Yes} 0.8  0.8

                          1.0         1.000000 4
```

```
[2]  {}              => {item6=No} 0.8   0.8     1.0     1.000000 4
[3]  {}              => {item5=No} 0.8   0.8     1.0     1.000000 4
[4]  {item5=Yes}        =>             0.2   1.0     0.2     2.500000 1
                        {item2=Yes}
[5]  {item5=Yes}        =>             0.2   1.0     0.2     2.500000 1
                        {item3=Yes}
[6]  {item5=Yes}        => {item4=No}  0.2   1.0     0.2     2.500000 1
[7]  {item5=Yes}     => {item1= Yes}   0.2   1.0     0.2     1.250000 1
[8]  {item5=Yes}        => {item6=No}  0.2   1.0     0.2     1.250000 1
[9]  {item3= No}     => {item6= Yes}   0.2   1.0     0.2     5.000000 1
[10]       {item6=    => {item3= No}   0.2   1.0     0.2   5.000000 1
              Yes}
[11] {item3=          => {item2=No}    0.2   1.0     0.2   2.500000 1
     No}
[12] {item3=          => {item4=Yes}   0.2   1.0     0.2   1.666667 1
     No}
[13] {item3=          => {item1= Yes}  0.2   1.0     0.2   1.250000 1
     No}
[14] {item3=          => {item5=No}    0.2   1.0     0.2   1.250000 1
     No}
[15]       {item6=    => {item2=No}    0.2   1.0     0.2   2.500000 1
              Yes}
[16]       {item6=    => {item4=Yes}   0.2   1.0     0.2   1.666667 1
              Yes}
[17]       {item6=    => {item1= Yes}  0.2   1.0     0.2   1.250000 1
              Yes}
[18]       {item6=    => {item5=No}    0.2   1.0     0.2   1.250000 1
              Yes}
[19] {item1=          => {item2= Yes}  0.2   1.0     0.2   5.000000 1
     No}
[20]       {item2=    => {item1= No}   0.2   1.0     0.2   5.000000 1
              Yes}
[21] {item1=          => {item3=yes}   0.2   1.0     0.2   2.500000 1
     No}
[22] {item1=          => {item4=Yes}   0.2   1.0     0.2   1.666667 1
     No}
[23] {item1=          => {item6=No}    0.2   1.0     0.2   1.250000 1
     No}
[24] {item1=          => {item5=No}    0.2   1.0     0.2   1.250000 1
```

```
        No}

   [25]      {item2=      => {item3=yes}  0.2    1.0      0.2   2.500000 1
                Yes}
   [26]      {item2=      => {item4=Yes}  0.2    1.0      0.2   1.666667 1
                Yes}
   [27]      {item2=      => {item6=No}   0.2    1.0      0.2   1.250000 1
                Yes}
   [28]      {item2=      => {item5=No}   0.2    1.0      0.2   1.250000 1
                Yes}
```

```
[29] {item2=Yes}        => {item3=Yes}  0.4 1.0     0.4    2.500000 2
[30] {item3=Yes}        => {item2=Yes}  0.4 1.0     0.4    2.500000 2
[31] {item2=Yes}        => {item4=No}   0.4 1.0     0.4    2.500000 2
[32] {item4=No}         => {item2=Yes}  0.4 1.0     0.4    2.500000 2
[33] {item2=Yes}        => {item1= Yes} 0.4 1.0     0.4    1.250000 2
[34] {item2=Yes}        => {item6=No}   0.4 1.0     0.4    1.250000 2
[35] {item3=Yes}        => {item4=No}   0.4 1.0     0.4    2.500000 2
[36] {item4=No}         => {item3=Yes}  0.4 1.0     0.4    2.500000 2
[37] {item3=Yes}        => {item1= Yes} 0.4 1.0     0.4    1.250000 2
[38] {item3=Yes}        => {item6=No}   0.4 1.0     0.4    1.250000 2
[39] {item4=No}         => {item1= Yes} 0.4 1.0     0.4    1.250000 2
[40] {item4=No}         => {item6=No}   0.4 1.0     0.4    1.250000 2
[41] {item2=No}         => {item4=Yes}  0.4 1.0     0.4    1.666667 2
[42] {item2=No}         => {item1= Yes} 0.4 1.0     0.4    1.250000 2
[43] {item2=No}         => {item5=No}   0.4 1.0     0.4    1.250000 2
[44] {item3=yes}        => {item4=Yes}  0.4 1.0     0.4    1.666667 2
[45] {item3=yes}        => {item6=No}   0.4 1.0     0.4    1.250000 2
[46] {item3=yes}        => {item5=No}   0.4 1.0     0.4    1.250000 2
[47] {item4=Yes}        => {item5=No}   0.6 1.0     0.6    1.250000 3
[48] {item2=Yes,item5=Yes} => {item3=Yes}    1.0     0.2    2.500000
     0.2                                             1
[49] {item3=Yes,item5=Yes} => {item2=Yes}    1.0     0.2    2.500000
     0.2                                             1
[50] {item2=Yes,item5=Yes} => {item4=No}     1.0     0.2    2.500000
     0.2                                             1
[51] {item4=No,item5=Yes}=> {item2=Yes}      1.0     0.2    2.500000
     0.2                                             1
[52] {item2=Yes,item5=Yes} => {item1= Yes}   1.0     0.2    1.250000
     0.2                                             1
[53] {item1= Yes,item5=Yes} => {item2=Yes}   1.0     0.2    2.500000
     0.2                                             1
[54] {item2=Yes,item5=Yes} => {item6=No}     1.0     0.2    1.250000
```

```
     0.2                                        1

[55] {item5=Yes,item6=No}=> {item2=Yes}    1.0  0.2   2.500000
     0.2                                        1
```

```
[56] {item3=Yes,item5=Yes}   => {item4=No}  0.2  1.0  0.2  2.500000 1
[57] {item4=No,item5=Yes}    => {item3=Yes} 0.2  1.0  0.2  2.500000 1
[58] {item3=Yes,item5=Yes}   => {item1= Yes} 0.2 1.0  0.2  1.250000 1
[59] {item1= Yes,item5=Yes}  => {item3=Yes} 0.2  1.0  0.2  2.500000 1
[60] {item3=Yes,item5=Yes}   => {item6=No}  0.2  1.0  0.2  1.250000 1
[61] {item5=Yes,item6=No}    => {item3=Yes} 0.2  1.0  0.2  2.500000 1
[62] {item4=No,item5=Yes}    => {item1= Yes} 0.2 1.0  0.2  1.250000 1
[63] {item1= Yes,item5=Yes}  => {item4=No} 0.2   1.0  0.2  2.500000 1
[64] {item4=No,item5=Yes}    => {item6=No} 0.2   1.0  0.2  1.250000 1
[65] {item5=Yes,item6=No}    => {item4=No} 0.2   1.0  0.2  2.500000 1
[66] {item1= Yes,item5=Yes}  => {item6=No}  0.2  1.0  0.2  1.250000 1
[67] {item5=Yes,item6=No}    => {item1= Yes} 0.2 1.0  0.2  1.250000 1
[68] {item3= No,item6= Yes}  => {item2=No}  0.2  1.0  0.2  2.500000 1
[69] {item2=No,item3= No}    => {item6= Yes} 0.2 1.0  0.2  5.000000 1
[70] {item2=No,item6= Yes}   => {item3= No} 0.2  1.0  0.2  5.000000 1
[71] {item3= No,item6= Yes}  => {item4=Yes} 0.2  1.0  0.2  1.666667 1
[72] {item3= No,item4=Yes}   => {item6= Yes} 0.2 1.0  0.2  5.000000 1
[73] {item4=Yes,item6= Yes}  => {item3= No} 0.2  1.0  0.2  5.000000 1
[74] {item3= No,item6= Yes}  => {item1= Yes} 0.2 1.0  0.2  1.250000 1
[75] {item1= Yes,item3= No}  => {item6= Yes} 0.2 1.0  0.2  5.000000 1
[76] {item1= Yes,item6= Yes} => {item3= No} 0.2  1.0  0.2  5.000000 1
```

```
[77]  {item3= No,item6=    => {item5=No}  0.2  1.0    0.2   1.250000 1
            Yes}
[78]      {item3=          => {item6= Yes}     1.0    0.2   5.000000 1
       No,item5=No}        0.2
[79]  {item5=No,item6=     => {item3= No} 0.2  1.0    0.2   5.000000 1
            Yes}
[80]  {item2=No,item3=     => {item4=Yes}      1.0    0.2   1.666667 1
            No}            0.2
[81]      {item3=          => {item2=No}  0.2  1.0    0.2   2.500000 1
       No,item4=Yes}
[82]  {item2=No,item3=     => {item1= Yes}     1.0    0.2   1.250000 1
            No}            0.2
```

```
[83]  {item1= Yes,item3=    => {item2=No} 0.2   1.0    0.2    2.500000 1
             No}
[84]   {item2=No,item3=     =>            0.2   1.0    0.2    1.250000 1
            No}             {item5=No}
[85]        {item3=         =>            0.2   1.0    0.2    2.500000 1
       No,item5=No}         {item2=No}
[86]        {item3=         => {item1=    0.2   1.0    0.2    1.250000 1
      No,item4=Yes}         Yes}
[87]  {item1= Yes,item3=    =>            0.2   1.0    0.2    1.666667 1
             No}            {item4=Yes}
[88]        {item3=         => {item5=No} 0.2   1.0    0.2    1.250000
      No,item4=Yes}                                           1
[89]        {item3=         =>            0.2   1.0    0.2    1.666667 1
       No,item5=No}         {item4=Yes}
[90]  {item1= Yes,item3=    => {item5=No} 0.2   1.0    0.2    1.250000 1
             No}
[91]        {item3=         => {item1=    0.2   1.0    0.2    1.250000
       No,item5=No}         Yes}                              1
[92]   {item2=No,item6=     =>            0.2   1.0    0.2    1.666667 1
            Yes}            {item4=Yes}

[93] {item4=Yes,item6= Yes} => {item2=No}       1.0    0.2    2.500000 1
0.2
[94] {item2=No,item6= Yes} => {item1= Yes}      1.0    0.2    1.250000 1
0.2
[95] {item1= Yes,item6= Yes} => {item2=No}      1.0    0.2    2.500000 1
0.2
[96] {item2=No,item6= Yes} => {item5=No} 0.2    1.0    0.2    1.250000 1

[97] {item5=No,item6= Yes} => {item2=No} 0.2    1.0    0.2    2.500000 1

[98] {item4=Yes,item6= Yes} => {item1= Yes}     1.0    0.2    1.250000 1
0.2
[99] {item1= Yes,item6= Yes} => {item4=Yes}     1.0    0.2    1.666667 1
0.2
[100] {item4=Yes,item6= Yes} => {item5=No}      1.0    0.2   1.250000 1
0.2
[101] {item5=No,item6= Yes} => {item4=Yes}      1.0    0.2   1.666667 1
0.2
[102] {item1= Yes,item6= Yes} => {item5=No}     1.0    0.2   1.250000 1
0.2
[103] {item5=No,item6= Yes} => {item1= Yes}     1.0    0.2   1.250000 1
0.2
```

```
[104] {item1= No,item2= Yes} => {item3=yes}    1.0      0.2   2.500000 1
0.2
[105] {item1= No,item3=yes} => {item2= Yes}    1.0      0.2   5.000000 1
0.2
[106] {item2= Yes,item3=yes} => {item1= No}    1.0      0.2   5.000000 1
0.2
[107] {item1= No,item2= Yes} => {item4=Yes}    1.0      0.2   1.666667 1
0.2
[108] {item1= No,item4=Yes} => {item2= Yes}    1.0      0.2    5.000000 1
0.2
[109] {item2= Yes,item4=Yes} => {item1= No}    1.0      0.2   5.000000 1
0.2
```

```
[110] {item1= No,item2=    => {item6=No}0.2   1.0    0.2   1.250000 1
Yes}
[111] {item1=              => {item2=    0.2   1.0    0.2   5.000000 1
No,item6=No}               Yes}
[112] {item2=              => {item1=    0.2   1.0    0.2   5.000000 1
Yes,item6=No}              No}
[113] {item1= No,item2=    => {item5=No}0.2   1.0    0.2   1.250000 1
Yes}
[114] {item1=              => {item2=    0.2   1.0    0.2   5.000000 1
No,item5=No}               Yes}
[115] {item2=              => {item1=    0.2   1.0    0.2   5.000000 1
Yes,item5=No}              No}
[116] {item1=              =>            0.2   1.0    0.2   1.666667 1
No,item3=yes}              {item4=Yes}
[117] {item1=              =>            0.2   1.0    0.2   2.500000 1
No,item4=Yes}              {item3=yes}
[118] {item1=              => {item6=No}0.2   1.0    0.2   1.250000 1
No,item3=yes}
[119] {item1=              =>            0.2   1.0    0.2   2.500000 1
No,item6=No}               {item3=yes}
[120] {item1=              => {item5=No}0.2   1.0    0.2   1.250000 1
No,item3=yes}
[121] {item1=              =>            0.2   1.0    0.2   2.500000 1
No,item5=No}               {item3=yes}
[122] {item1=              => {item6=No}0.2   1.0    0.2   1.250000 1
No,item4=Yes}
[123] {item1=              =>            0.2   1.0    0.2   1.666667 1
No,item6=No}               {item4=Yes}
[124] {item1=              => {item5=No}0.2   1.0    0.2   1.250000 1
No,item4=Yes}
[125] {item1=              =>            0.2   1.0    0.2   1.666667 1
No,item5=No}               {item4=Yes}
 [ reached 'max' / getOption("max.print") -- omitted 450 rows ]
```

```
> rules<-
apriori(shopping,parameter=list(support=0.5,conf=0.7))
Apriori


Parameter specification:
 confidence minval smax arem aval
```

```
      0.7 0.1 1 none FALSE
originalSupport maxtime support minlen
          maxlen TRUE      5    0.5      1          10
target ext
 rules
 TRUE
```

```
Algorithmic control:
 filter tree heap memopt load sort verbose
   0.1 TRUE TRUE FALSE TRUE 2 TRUE


Absolute minimum support count: 2


set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[12 item(s), 5 transaction(s)] done
[0.00s]. sorting and recoding items ... [6 item(s)] done
[0.00s]. creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done
[0.00s]. writing ... [31 rule(s)] done
[0.00s]. creating S4 object ... done
[0.00s].
Warning message:
Column(s) 1, 2, 3, 4, 5, 6 not logical or factor. Applying default discretization (see '?
discretizeDF').
> inspect (rules)
    lhs          rhs       support confidence coverage lift count
[1]  {}         => {item1=yes}  0.8     0.80     1.0 1.0000   4
[2]  {}         => {item3=yes}  0.8     0.80     1.0 1.0000   4
[3]  {}         => {item6=no}   0.8     0.80     1.0 1.0000   4
[4]  {}         => {item5=no}   0.8     0.80     1.0 1.0000   4
[5]  {item2=yes} =>                 0.6     1.00     0.6 1.2500   3
     {item3=yes}
[6]  {item3=yes} =>                 0.6     0.75     0.8 1.2500   3
     {item2=yes}
[7]  {item2=yes} => {item6=no}  0.6     1.00     0.6 1.2500   3
[8]  {item6=no} => {item2=yes}  0.6     0.75     0.8 1.2500   3
[9]  {item4=yes} => {item5=no}  0.6     1.00     0.6 1.2500   3
[10] {item5=no} =>                  0.6     0.75        0.8         3
{item4=yes}                                              1.2500
```

```
[11] {item1=yes} =>              0.6    0.75       0.8        3
{item3=yes}                                      0.9375
[12] {item3=yes} =>              0.6    0.75       0.8        3
{item1=yes}                                      0.9375
```

```
[13] {item1=yes} =>              0.6    0.75   0.8 0.9375    3
{item6=no}
[14] {item6=no} =>               0.6    0.75   0.8 0.9375    3
{item1=yes}
[15] {item1=yes} =>              0.6    0.75   0.8 0.9375    3
{item5=no}
[16] {item5=no} =>               0.6    0.75   0.8 0.9375    3
{item1=yes}
[17] {item3=yes} =>              0.8    1.00   0.8 1.2500    4
{item6=no}
[18] {item6=no} =>               0.8    1.00   0.8 1.2500    4
{item3=yes}
[19] {item3=yes} =>              0.6    0.75   0.8 0.9375    3
{item5=no}
[20] {item5=no} =>               0.6    0.75   0.8 0.9375    3
{item3=yes}
[21] {item6=no} => {item5=no}   0.6    0.75   0.8 0.9375    3

[22] {item5=no} => {item6=no}   0.6    0.75   0.8 0.9375    3

[23] {item2=yes,

     item3=yes} => {item6=no}  0.6     1.00    0.6 1.2500   3

[24] {item2=yes,

     item6=no}   => {item3=yes} 0.6     1.00    0.6 1.2500   3

[25] {item3=yes,

     item6=no}   => {item2=yes} 0.6     0.75    0.8 1.2500   3

[26] {item1=yes,

     item3=yes}  => {item6=no}  0.6     1.00    0.6 1.2500   3

[27] {item1=yes,

     item6=no}   => {item3=yes} 0.6     1.00    0.6 1.2500   3

[28] {item3=yes,

     item6=no}   => {item1=yes} 0.6     0.75    0.8 0.9375   3

[29] {item3=yes,

     item6=no}   => {item5=no}  0.6     0.75    0.8 0.9375   3

[30] {item3=yes,

     item5=no}   => {item6=no}  0.6     1.00    0.6 1.2500   3

[31] {item5=no,

     item6=no}   => {item3=yes} 0.6     1.00    0.6 1.2500   3
```

```
> rules<- apriori(shopping,parameter=list(minlen=2, maxlen=5, support=0.5,conf=0.7))
```

```
Apriori

Parameter specification:
 confidence minval smax arem aval
      0.7 0.1 1 none FALSE
 originalSupport maxtime support minlen
         maxlen TRUE      5     0.5     2          5
 target ext
  rules
  TRUE

Algorithmic control:
 filter tree heap memopt load sort verbose
   0.1 TRUE TRUE FALSE TRUE 2 TRUE

Absolute minimum support count: 2

set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[12 item(s), 5 transaction(s)] done
[0.00s]. sorting and recoding items ... [6 item(s)] done
[0.00s]. creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 done
[0.00s]. writing ... [27 rule(s)] done
[0.00s]. creating S4 object ... done
[0.00s].
Warning message:
Column(s) 1, 2, 3, 4, 5, 6 not logical or factor. Applying default discretization (see '?
discretizeDF').
> inspect(rules)
    lhs          rhs       support confidence coverage lift count
[1] {item2=yes} =>                  0.6      1.00   0.6 1.2500    3
{item3=yes}
```

```
[2] {item3=yes} =>           0.6    0.75   0.8 1.2500    3
{item2=yes}
```

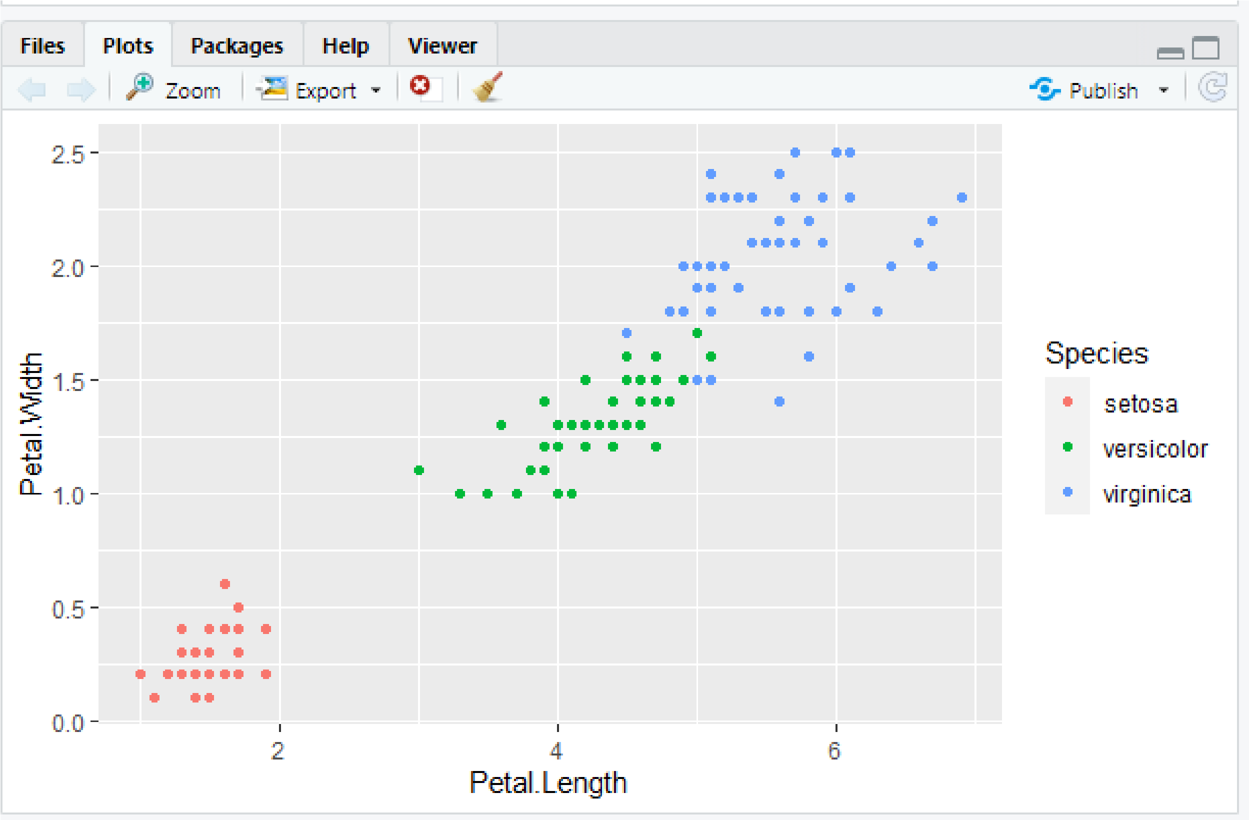
```
[3] {item2=yes} => {item6=no}   0.6     1.00   0.6 1.2500    3
[4] {item6=no} => {item2=yes}   0.6     0.75   0.8 1.2500    3
[5] {item4=yes} => {item5=no}   0.6     1.00   0.6 1.2500    3
[6] {item5=no} => {item4=yes}   0.6     0.75   0.8 1.2500    3
[7] {item1=yes} =>              0.6     0.75   0.8 0.9375    3
{item3=yes}
[8] {item3=yes} =>              0.6     0.75   0.8 0.9375    3
{item1=yes}
[9] {item1=yes} => {item6=no}   0.6     0.75   0.8 0.9375    3
[10] {item6=no} =>              0.6     0.75   0.8 0.9375    3
{item1=yes}
[11] {item1=yes} =>             0.6     0.75   0.8 0.9375    3
{item5=no}
[12] {item5=no} =>              0.6     0.75   0.8 0.9375    3
{item1=yes}
[13] {item3=yes} =>             0.8     1.00   0.8 1.2500    4
{item6=no}
[14] {item6=no} =>              0.8     1.00   0.8 1.2500    4
{item3=yes}
[15] {item3=yes} =>             0.6     0.75   0.8 0.9375    3
{item5=no}
[16] {item5=no} =>              0.6     0.75   0.8 0.9375    3
{item3=yes}
[17] {item6=no} => {item5=no}   0.6     0.75   0.8 0.9375    3
[18] {item5=no} => {item6=no}   0.6     0.75   0.8 0.9375    3
[19] {item2=yes,
     item3=yes} => {item6=no}   0.6     1.00    0.6 1.2500   3
[20] {item2=yes,
     item6=no}   => {item3=yes} 0.6     1.00    0.6 1.2500   3
[21] {item3=yes,
     item6=no}   => {item2=yes} 0.6     0.75    0.8 1.2500   3
[22] {item1=yes,
     item3=yes}  => {item6=no}  0.6     1.00    0.6 1.2500   3
[23] {item1=yes,
     item6=no}   => {item3=yes} 0.6     1.00    0.6 1.2500   3
[24] {item3=yes,
     item6=no}   => {item1=yes} 0.6     0.75    0.8 0.9375  3
[25] {item3=yes,
     item6=no}   => {item5=no}  0.6      0.75     0.8 0.9375   3
[26] {item3=yes,
     item5=no}   => {item6=no}  0.6      1.00     0.6 1.2500   3
[27] {item5=no,
     item6=no}   => {item3=yes} 0.6      1.00     0.6 1.2500   3
```

10. Implementation and analysis of clustering algorithms like K-Means , Agglomerative

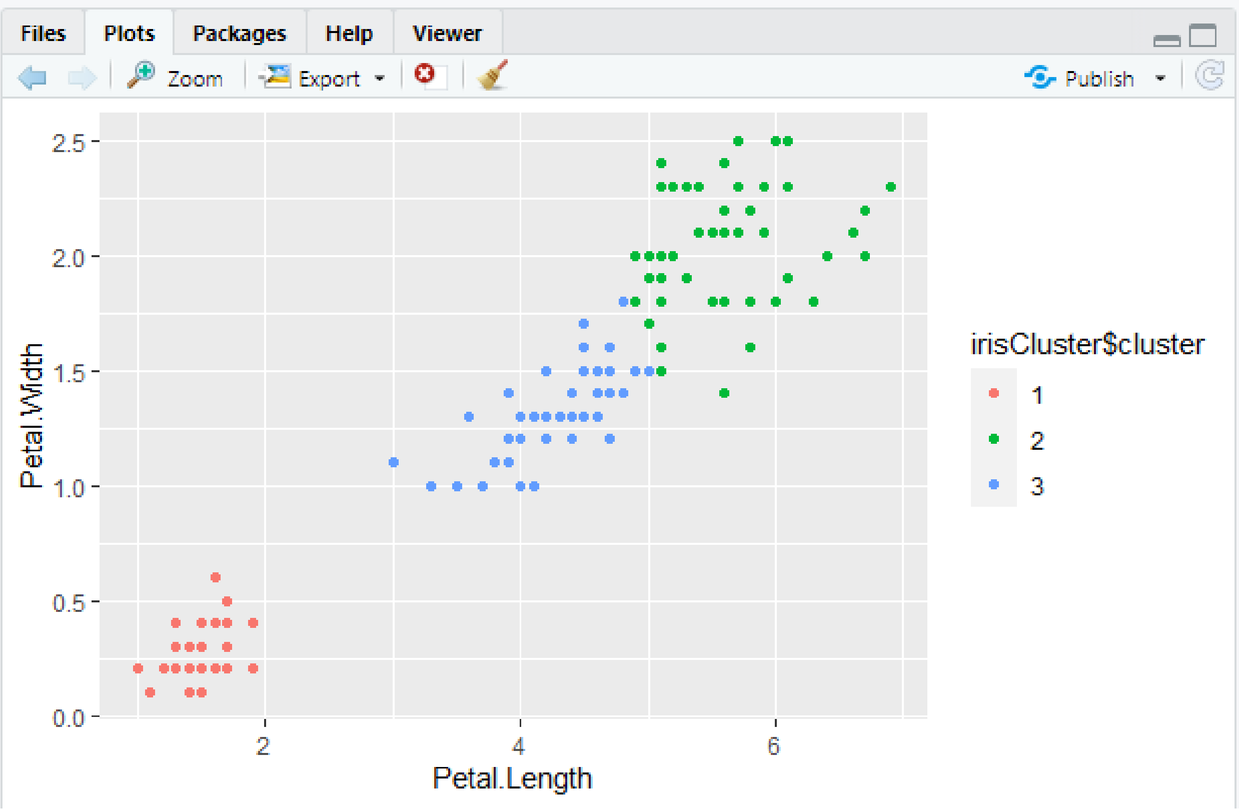
Source code:

K-MEANS

```
> setwd("C:/Users/admin/Desktop/R-prog")

> head(iris)

  Sepal.Length Sepal.Width Petal.Length

1          5.1         3.5          1.4

2          4.9         3.0          1.4

3          4.7         3.2          1.3

4          4.6         3.1          1.5

5          5.0         3.6          1.4

6          5.4         3.9          1.7

  Petal.Width Species

1         0.2  setosa

2         0.2  setosa

3         0.2  setosa

4         0.2  setosa

5         0.2  setosa

6         0.4  setosa

> library(ggplot2)

> ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom_point()
```

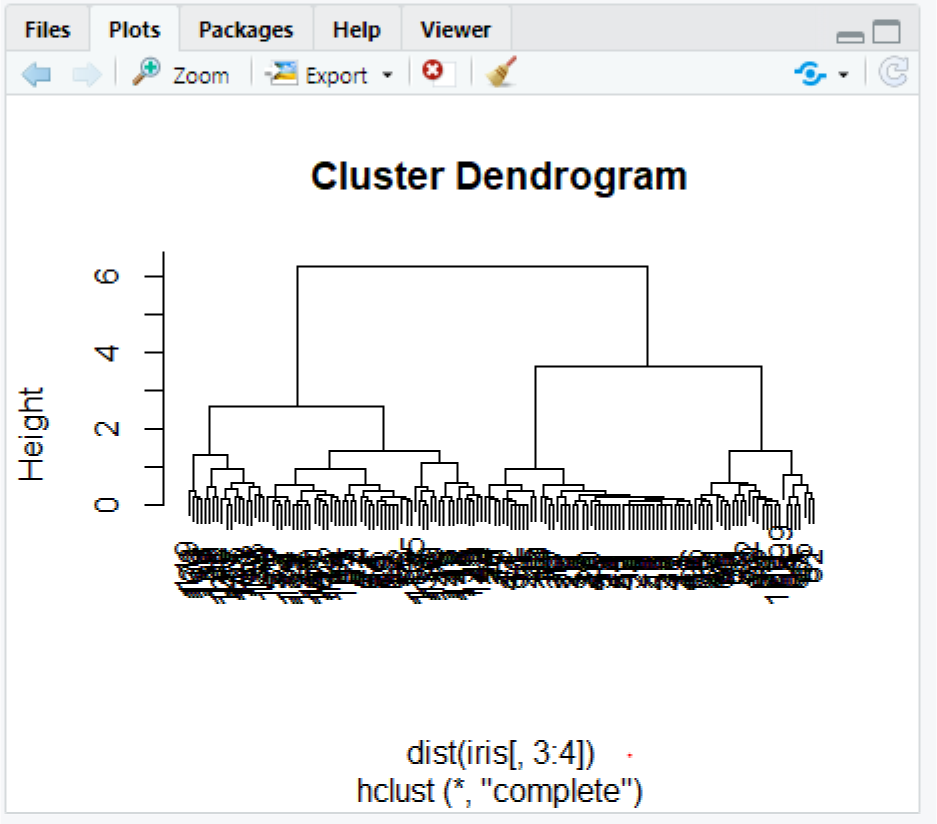


```
> set.seed(20)
> risCluster <- kmeans(iris[, 3:4], 3, nstart = 20)
> irisCluster <- kmeans(iris[, 3:4], 3, nstart = 20)
> irisCluster
K-means clustering with 3 clusters of sizes 50, 48, 52

Cluster means:
  Petal.Length Petal.Width
1     1.462000    0.246000
2     5.595833    2.037500
3     4.269231    1.342308

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [22] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [43] 1 1 1 1 1 1 1 1 3 3 3 3 3 3 3 3 3 3 3 3 3
 [64] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 2
 [85] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 2 2 2 2 2
[106] 2 3 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2
[127] 3 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2
[148] 2 2 2

Within cluster sum of squares by cluster:
[1] 2.02200 16.29167 13.05769
 (between_SS / total_SS = 94.3 %)

Available components:

[1] "cluster"      "centers"      "totss"
[4] "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
> table(irisCluster$cluster, iris$Species)

    setosa versicolor virginica
  1     50          0         0
  2      0          2        46
  3      0         48         4
> irisCluster$cluster <- as.factor(irisCluster$cluster)
> ggplot(iris, aes(Petal.Length, Petal.Width, color = irisCluster$cluster)) +
geom_point()
```

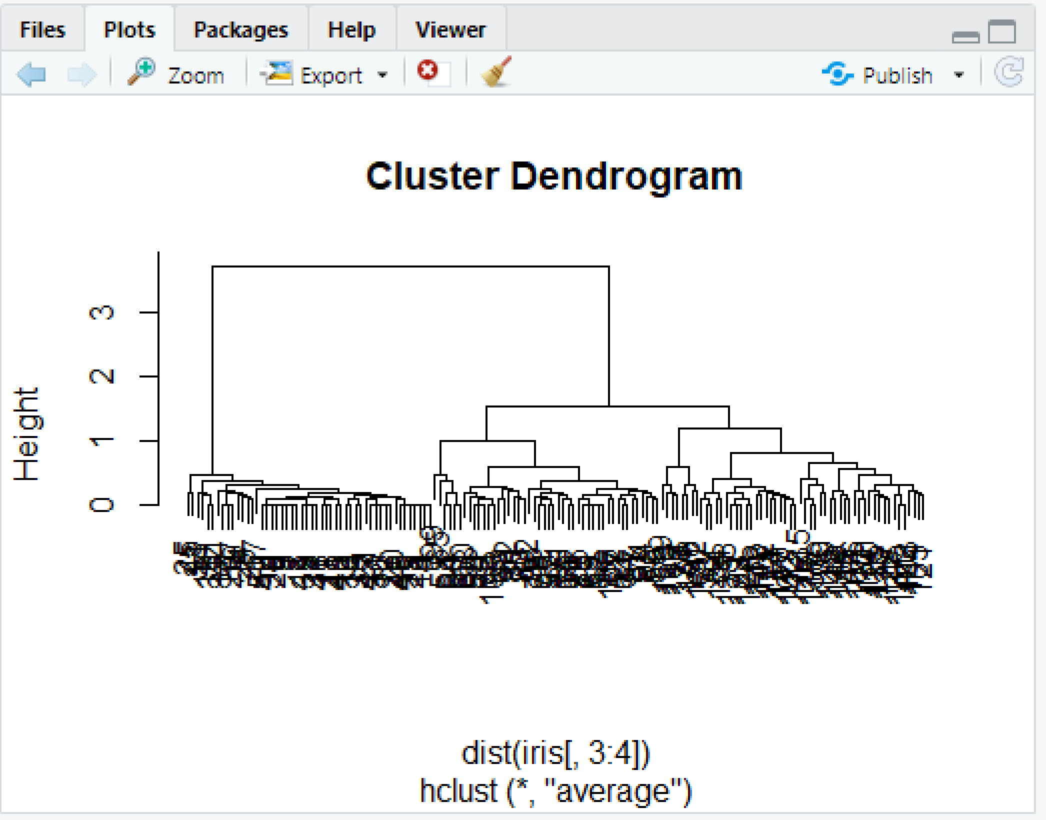

Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
2.5
2.0
1.5
1.0
0.5
0.0
Petal.Width
2
4
6
Petal.Length
irisCluster$cluster
1
2
3

Agglomerative Clustering

```
> head(iris)
Sepal.Length Sepal.Width Petal.Length Petal.Width
1        5.1        3.5        1.4        0.2
2        4.9        3.0        1.4        0.2
3        4.7        3.2        1.3        0.2
4        4.6        3.1        1.5        0.2
5        5.0        3.6        1.4        0.2
6        5.4        3.9        1.7        0.4
Species
1  setosa
2  setosa
3  setosa
4  setosa
5  setosa
6  setosa
> clusters <- hclust(dist(iris[, 3:4]))
> plot(clusters)
```



```
> clusterCut<- cutree(clusters, 3)

> table(clusterCut, iris$Species)

clusterCut setosa  versicolor virginica

1 50            0        0

2      0        21       50

3      0        29        0

> clusters <- hclust(dist(iris[,3:4]), method = 'average')

> plot(clusters)
```

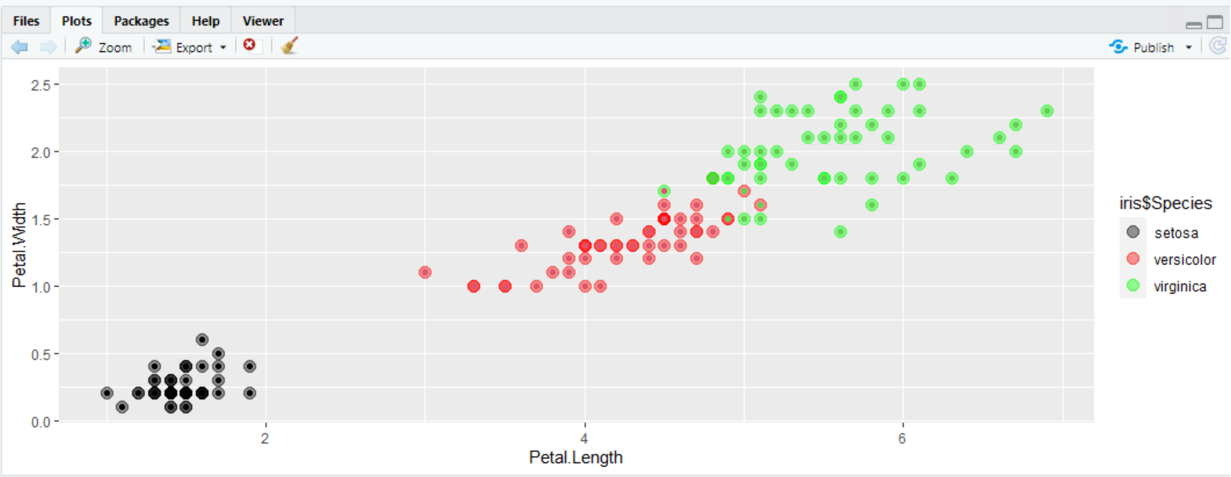


```
>clusterCut <- cutree(clusters, 3)

>table(clusterCut, iris$Species)


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

```
>ggplot(iris, aes(Petal.Length, Petal.Width, color = iris$Species)) +       geom_point(alpha
+                                              = 0.4, size = 3.5) + geom_point(col =
        clusterCut) + scale_color_manual(values = c('black', 'red', 'green'))
```

Warning message:

Use of `iris$Species` is discouraged. Use `Species` instead.


Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Petal.Width
2.5
2.0
1.5
1.0
0.5
0.0
2
4
6
Petal.Length
iris$Species
setosa
versicolor
virginica