ADBMS PRACTICAL JOURNAL Saiganesh

29/08/2023: Content Beyond Syllabus

**CREATE TABLE EMPLOYEE29 (emp\_no INTEGER, emp\_name VARCHAR(15), salary INTEGER);**

Table created.

**INSERT INTO EMPLOYEE29 VALUES (101, 'Alex', 65000);**

1 row created

**INSERT INTO EMPLOYEE29 VALUES (&EMP\_NO, '&EMP\_NAME', &SALARY);**

Enter value for emp\_no: 102

Enter value for emp\_name: Adam

Enter value for salary: 56000

**SELECT \* FROM EMPLOYEE29;**

EMP\_NO EMP\_NAME SALARY

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

101 Alex 65000

102 Adam 56000

103 Eve 58000

104 Phil 48000

105 Claire 59000

106 Joe 100000

**INSERT ALL**

**2 INTO EMPLOYEE29 VALUES (107,'Gloria', 150000)**

**3 INTO EMPLOYEE29 VALUES (108,'Luke', 20000)**

**4 SELECT \* FROM EMPLOYEE29;**

**SELECT \* EMPLOYEE29;**

EMP\_NO EMP\_NAME SALARY

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

107 Gloria 150000

108 Luke 20000

**ALTER TABLE EMPLOYEE29 ADD comm INTEGER;**

Table altered.

**SELECT \* FROM EMPLOYYE29;**

EMP\_NO EMP\_NAME SALARY COMM

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

101 Alex 65000

102 Adam 56000

103 Eve 58000

104 Phil 48000

105 Claire 59000

106 Joe 100000

107 Gloria 150000

**CREATE TABLE empone (emp\_no INTEGER, salary INTEGER);**

Table created.

**ALTER TABLE empone ADD CONSTRAINT pk1 PRIMARY KEY(emp\_no);**

Table altered.

**ALTER TABLE empone ADD CONSTRAINT CHECK1 CHECK(salary>25000 AND salary<=10000);**

Table altered.

PRACTICAL 1: Range Partitioning Date: 31/08/2023

Note: Range partitioning happens on numeric columns

Creating Partitions:

* **CREATE TABLE sal\_range101(salesman\_id NUMBER(5), salesman\_name VARCHAR2(30), sales\_amount NUMBER(10), sales\_date DATE)**

**2 PARTITION BY RANGE (sales\_date)(**

**3 PARTITION sales\_jan2000 VALUES LESS THAN (TO\_DATE('01/02/2000','DD/MM/YYYY')),**

**4 PARTITION sales\_feb2000 VALUES LESS THAN (TO\_DATE('01/03/2000','DD/MM/YYYY')),**

**5 PARTITION sales\_march2000 VALUES LESS THAN (TO\_DATE ('01/04/2000', 'DD/MM/YYYY')),**

**6 PARTITION sales\_apr2000 VALUES LESS THAN (TO\_DATE ('01/05/2000','DD/MM/YYYY'))**

**7 );**

Table created.

To check if partitions are built correctly:

**SELECT TABLE\_NAME, PARTITION\_NAME FROM USER\_TAB\_PARTITIONS WHERE TABLESPACE\_NAME = 'USERS';**

TABLE\_NAME PARTITION\_NAME

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

SAL\_RANGE101 SALES\_JAN2000

SAL\_RANGE101 SALES\_FEB2000

SAL\_RANGE101 SALES\_MARCH2000

SAL\_RANGE101 SALES\_APR2000

Insert date in sal\_range101 table:

**INSERT INTO sal\_range101 VALUES( 1,'James Bond',5000,TO\_DATE('23/02/2000','DD/MM/YYYY'));**

1 row created.

Display Data from sal\_range101 table:

**SELECT \* FROM sal\_range101;**

SALESMAN\_ID SALESMAN\_NAME SALES\_AMOUNT SALES\_DAT

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

4 Samuel 4500 09-JAN-00

6 Indira 4300 24-JAN-00

8 Rahul 8920 15-JAN-00

1 James Bond 5000 23-FEB-00

10 Robert 8920 15-FEB-00

3 Sasha 4000 12-MAR-00

5 Scarlett 4300 14-MAR-00

9 Priyanka 8920 15-MAR-00

2 Margaret 5000 23-APR-00

7 Rajiv 8920 24-APR-00

10 rows selected.

Another Example:

**CREATE TABLE emp101(emp\_no INTEGER PRIMARY KEY, emp\_name VARCHAR2(10),**

**2 salary INTEGER)**

**3 PARTITION BY RANGE(emp\_no)(**

**4 PARTITION E1 VALUES LESS THAN (103),**

**5 PARTITION E2 VALUES LESS THAN (104)**

**6 );**

Table created.

Check to see if the partitions are correctly built:

**SELECT TABLE\_NAME, PARTITION\_NAME FROM USER\_TAB\_PARTITIONS WHERE TABLESPACE\_NAME = 'USERS';**

TABLE\_NAME PARTITION\_NAME

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

EMP101 E1

EMP101 E2

**INSERT INTO emp101 VALUES(102,'SITA',31000);**

1 row created.

**SELECT \* FROM EMP101;**

EMP\_NO EMP\_NAME SALARY

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

101 RAM 25000

102 SITA 31000

103 RAVAN 35000

**SELECT \* FROM EMP101 PARTITION(E1);**

EMP\_NO EMP\_NAME SALARY

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

101 RAM 25000

102 SITA 31000

**SELECT \* FROM EMP101 PARTITION(E2);**

EMP\_NO EMP\_NAME SALARY

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

103 RAVAN 35000

List Partitioning:

When a column can be grouped based on a category. Usually a column with VARCHAR2 datatype.

**CREATE TABLE SAL\_REGION101(**

**2 product\_id NUMBER(6),**

**3 sales\_date DATE,**

**4 state VARCHAR2(30))**

**5 PARTITION BY LIST (state)(**

**6 PARTITION region\_north VALUES('Himachal Pradesh','Haryana','Punjab'),**

**7 PARTITION region\_south VALUES('Karnataka','Tamil Nadu','Telangana'),**

**8 PARTITION region\_east VALUES('West Bengal','Odisha','Jharkhand'),**

**9 PARTITION region\_west VALUES('Goa','Gujarat','Maharshtra'),**

**10 PARTITION region\_other VALUES(DEFAULT))**

**11 enable row movement;**

Table created.

**SELECT TABLE\_NAME, PARTITION\_NAME FROM USER\_TAB\_PARTITIONS WHERE TABLESPACE\_NAME = 'USERS';**

TABLE\_NAME PARTITION\_NAME

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

SAL\_REGION101 REGION\_NORTH

SAL\_REGION101 REGION\_SOUTH

SAL\_REGION101 REGION\_EAST

SAL\_REGION101 REGION\_WEST

SAL\_REGION101 REGION\_OTHER

**INSERT INTO sal\_region101 VALUES(12,TO\_DATE('14/04/2000','DD/MM/YYYY'),'Maharashtra');**

1 row created.

**SELECT \* FROM sal\_region101;**

PRODUCT\_ID SALES\_DAT STATE

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

12 12-DEC-00 Himachal Pradesh

12 12-JAN-00 Telangana

14 14-NOV-00 West Bengal

12 14-APR-00 Goa

12 14-APR-00 Uttarakhand

12 14-APR-00 Maharashtra

6 rows selected.

**SELECT \* FROM sal\_region101 PARTITION(region\_west);**

PRODUCT\_ID SALES\_DAT STATE

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

12 14-APR-00 Goa

**SELECT \* FROM sal\_region101 PARTITION(region\_other);**

PRODUCT\_ID SALES\_DAT STATE

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

12 14-APR-00 Uttarakhand

12 14-APR-00 Maharashtra

Hash Partitioning: (Out of Syllabus – sometimes asked in practical exam)

Hash Partitioning is performed by system (Implicit); Best when there is no numeric column to partition by range or a column that can’t be grouped based on a category.

**CREATE TABLE sales\_hash101( salesman\_id NUMBER(5), salesman\_name VARCHAR2(30),sales\_amount NUMBER(10), week\_no NUMBER(2))**

**PARTITION BY HASH(salesman\_id)**

**PARTITIONS 4;**

Table created.

Check to see if the partitions are correctly built:

**SELECT TABLE\_NAME, PARTITION\_NAME FROM USER\_TAB\_PARTITIONS WHERE TABLESPACE\_NAME = 'USERS';**

TABLE\_NAME PARTITION\_NAME

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

SALES\_HASH101 SYS\_P21

SALES\_HASH101 SYS\_P22

SALES\_HASH101 SYS\_P23

SALES\_HASH101 SYS\_P24

4 rows selected.

**INSERT ALL**

**INTO sales\_hash101 VALUES(101,'John',44000,12)**

**INTO sales\_hash101 VALUES(102,'Alice',54000,13)**

**INTO sales\_hash101 VALUES(103,'Drake',64000,14)**

**INTO sales\_hash101 VALUES(104,'Robert',74000,15)**

**INTO sales\_hash101 VALUES(105,'Julie',94000,13)**

**INTO sales\_hash101 VALUES(106,'James',34000,14)**

**INTO sales\_hash101 VALUES(107,'Kennedy',24000,12)**

**SELECT \* FROM DUAL;**

7 rows created.

**SELECT \* FROM sales\_hash101;**

SALESMAN\_ID SALESMAN\_NAME SALES\_AMOUNT WEEK\_NO

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

104 Robert 74000 15

102 Alice 54000 13

103 Drake 64000 14

105 Julie 94000 13

107 Kennedy 24000 12

101 John 44000 12

106 James 34000 14

7 rows selected.

Practical 2: Execution of OLAP & OLTP

Analytical Function:

**CREATE TABLE ssb101 ( emp\_no INTEGER, dep\_no INTEGER, bdate DATE, sal INTEGER, comm INTEGER, job VARCHAR2(10));**

Table created.

**INSERT ALL**

**INTO ssb101 VALUES(101,10,TO\_DATE('12-JAN-82','DD-MON-YY'),22000,1000,'CLERK')**

**INTO ssb101 VALUES(102,10,TO\_DATE('13-FEB-83','DD-MON-YY'),33000,2000,'CLERK')**

**INTO ssb101 VALUES(103,10,TO\_DATE('14-MAR-84','DD-MON-YY'),44000,200,'CLERK')**

**INTO ssb101 VALUES(104,20,TO\_DATE('15-APR-87','DD-MON-YY'),55000,3000,'MANAGER')**

**INTO ssb101 VALUES(105,20,TO\_DATE('14-JUN-82','DD-MON-YY'),38000,4500,'MANAGER')**

**INTO ssb101 VALUES(106,20,TO\_DATE('15-AUG-88','DD-MON-YY'),44000,500,'MANAGER')**

**INTO ssb101 VALUES(107,10,TO\_DATE('31-DEC-81','DD-MON-YY'),58000,8000,'MANAGER')**

**INTO ssb101 VALUES(108,20,TO\_DATE('25-JUL-88','DD-MON-YY'),49000,700,'CLERK')**

**INTO ssb101 VALUES(109,20,TO\_DATE('22-JUN-84','DD-MON-YY'),49500,2700,'CLERK')**

**INTO ssb101 VALUES(110,10,TO\_DATE('29-JAN-82','DD-MON-YY'),56000,1700,'MANAGER')**

**SELECT \* FROM DUAL;**

10 rows created.

**SELECT \* FROM ssb101;**

EMP\_NO DEP\_NO BDATE SAL 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

109 20 22-JUN-84 49500 2700 CLERK

110 10 29-JAN-82 56000 1700 MANAGER

10 rows selected.

ROLLUP:

**SELECT dep\_no, job,COUNT(\*), sum(SAL) FROM ssb101**

**2 GROUP BY ROLLUP(dep\_no,job);**

DEP\_NO JOB COUNT(\*) SUM(SAL)

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

10 CLERK 3 99000

10 MANAGER 2 114000

10 5 213000

20 CLERK 2 98500

20 MANAGER 3 137000

20 5 235500

10 448500

7 rows selected.

Partial ROLLUP:

**SELECT dep\_no, job, sum(SAL) FROM ssb101 WHERE dep\_no IN(10,20)**

**2 GROUP BY dep\_no, ROLLUP(job);**

DEP\_NO JOB SUM(SAL)

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

10 CLERK 99000

10 MANAGER 114000

10 213000

20 CLERK 98500

20 MANAGER 137000

20 235500

6 rows selected.

RANK ():

**SELECT emp\_no,dep\_no,sal,comm,RANK() OVER(PARTITION BY dep\_no ORDER BY sal) AS RANK**

**2 FROM ssb101;**

EMP\_NO DEP\_NO SAL COMM RANK

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

101 10 22000 1000 1

102 10 33000 2000 2

103 10 44000 200 3

110 10 56000 1700 4

107 10 58000 8000 5

105 20 38000 4500 1

106 20 44000 500 2

108 20 49000 700 3

109 20 49500 2700 4

104 20 55000 3000 5

10 rows selected.

DENSE\_RANK ():

**SELECT emp\_no,dep\_no,sal,comm,DENSE\_RANK() OVER(PARTITION BY dep\_no ORDER BY sal) AS RANK**

**2 FROM ssb101;**

EMP\_NO DEP\_NO SAL COMM RANK

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

101 10 22000 1000 1

102 10 33000 2000 2

103 10 44000 200 3

110 10 56000 1700 4

107 10 58000 8000 5

105 20 38000 4500 1

106 20 44000 500 2

108 20 49000 700 3

109 20 49500 2700 4

104 20 55000 3000 5

10 rows selected.

LEAD:

**SELECT emp\_no,bdate,LEAD(bdate,1) OVER(ORDER BY bdate) AS "next"**

**2 FROM ssb101;**

EMP\_NO BDATE next

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

107 31-DEC-81 12-JAN-82

101 12-JAN-82 29-JAN-82

110 29-JAN-82 14-JUN-82

105 14-JUN-82 13-FEB-83

102 13-FEB-83 14-MAR-84

103 14-MAR-84 22-JUN-84

109 22-JUN-84 15-APR-87

104 15-APR-87 25-JUL-88

108 25-JUL-88 15-AUG-88

106 15-AUG-88

10 rows selected.

LAG:

**SELECT emp\_no,bdate,LAG(bdate,1) OVER(ORDER BY bdate) AS "previous"**

**2 FROM ssb101;**

EMP\_NO BDATE previous

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

107 31-DEC-81

101 12-JAN-82 31-DEC-81

110 29-JAN-82 12-JAN-82

105 14-JUN-82 29-JAN-82

102 13-FEB-83 14-JUN-82

103 14-MAR-84 13-FEB-83

109 22-JUN-84 14-MAR-84

104 15-APR-87 22-JUN-84

108 25-JUL-88 15-APR-87

106 15-AUG-88 25-JUL-88

10 rows selected.

FIRST:

**SELECT dep\_no,sal,MAX(sal)KEEP(DENSE\_RANK FIRST ORDER BY sal DESC)**

**2 OVER(PARTITION BY dep\_no)"max"**

**3 FROM ssb101;**

DEP\_NO SAL max

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

10 22000 58000

10 44000 58000

10 58000 58000

10 33000 58000

10 56000 58000

20 49500 55000

20 49000 55000

20 44000 55000

20 55000 55000

20 38000 55000

10 rows selected.

LAST:

**SELECT dep\_no,sal,MIN(sal)KEEP(DENSE\_RANK LAST ORDER BY sal DESC)**

**2 OVER(PARTITION BY dep\_no)"min"**

**3 FROM ssb101;**

DEP\_NO SAL min

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

10 22000 22000

10 44000 22000

10 58000 22000

10 33000 22000

10 56000 22000

20 49500 38000

20 49000 38000

20 44000 38000

20 55000 38000

20 38000 38000

10 rows selected.

**CREATE TYPE type\_name101 AS OBJECT (**

**fname VARCHAR(20),**

**mname VARCHAR(20),**

**lname VARCHAR(20));**

**/**

Type created.

**CREATE TYPE typ\_address101 AS OBJECT(**

**street VARCHAR(20),**

**city VARCHAR(20),**

**pincode NUMBER(10));**

**/**

Type created.

**CREATE TABLE cust101 (**

**c\_id NUMBER(5) PRIMARY KEY,**

**c\_name type\_name101, 🡨 here c\_name is the object of typ\_name**

**c\_add typ\_address101,**

**c\_phno NUMBER(10));**

Type Created.

**INSERT INTO cust101 VALUES(1,type\_name101('John','F','Kennedy'),**

**typ\_address101('White House', 'Washington',123),1256468);**

1 row created.

**INSERT INTO cust101 VALUES(2,type\_name101('John','Q','Adams'),**

**typ\_address101('White House', 'Washington',123),0354688);**

1 row created.

**INSERT INTO cust101 VALUES(3,type\_name101('William','H','Harrison'),**

**typ\_address101('White House', 'Washington',123),13546846);**

1 row created.

**INSERT INTO cust101 VALUES(4,type\_name101('James','K','Polk'),**

**typ\_address101('White House', 'Washington',123),467765464);**

1 row created.

**INSERT INTO cust101 VALUES(5,type\_name101('Donald','J','Trump'),**

**typ\_address101('White House', 'Washington',123),13848463);**

1 row created.

**desc cust101;**

Name Null? Type

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

C\_ID NOT NULL NUMBER(5)

C\_NAME TYPE\_NAME101

C\_ADD TYP\_ADDRESS101

C\_PHNO NUMBER(10)

**set describe depth 2;**

**desc cust101;**

Name Null? Type

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

C\_ID NOT NULL NUMBER(5)

C\_NAME TYPE\_NAME101

FNAME VARCHAR2(20)

MNAME VARCHAR2(20)

LNAME VARCHAR2(20)

C\_ADD TYP\_ADDRESS101

STREET VARCHAR2(20)

CITY VARCHAR2(20)

PINCODE NUMBER(10)

C\_PHNO NUMBER(10)

**SELECT \* FROM cust101;**

C\_ID

----------

C\_NAME(FNAME, MNAME, LNAME)

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

C\_ADD(STREET, CITY, PINCODE)

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

C\_PHNO

----------

1

TYPE\_NAME101('John', 'F', 'Kennedy')

TYP\_ADDRESS101('White House', 'Washington', 123)

1256468

C\_ID

----------

C\_NAME(FNAME, MNAME, LNAME)

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

C\_ADD(STREET, CITY, PINCODE)

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

C\_PHNO

----------

2

TYPE\_NAME101('John', 'Q', 'Adams')

TYP\_ADDRESS101('White House', 'Washington', 123)

354688

C\_ID

----------

C\_NAME(FNAME, MNAME, LNAME)

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

C\_ADD(STREET, CITY, PINCODE)

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

C\_PHNO

----------

3

TYPE\_NAME101('William', 'H', 'Harrison')

TYP\_ADDRESS101('White House', 'Washington', 123)

13546846

C\_ID

----------

C\_NAME(FNAME, MNAME, LNAME)

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

C\_ADD(STREET, CITY, PINCODE)

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

C\_PHNO

----------

4

TYPE\_NAME101('James', 'K', 'Polk')

TYP\_ADDRESS101('White House', 'Washington', 123)

467765464

C\_ID

----------

C\_NAME(FNAME, MNAME, LNAME)

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

C\_ADD(STREET, CITY, PINCODE)

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

C\_PHNO

----------

5

TYPE\_NAME101('Donald', 'J', 'Trump')

TYP\_ADDRESS101('White House', 'Washington', 123)

13848463

**select c\_id from cust101;**

C\_ID

----------

1

2

3

4

5

SQL> **select c\_name from cust101;**

C\_NAME(FNAME, MNAME, LNAME)

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

TYPE\_NAME101('John', 'F', 'Kennedy')

TYPE\_NAME101('John', 'Q', 'Adams')

TYPE\_NAME101('William', 'H', 'Harrison')

TYPE\_NAME101('James', 'K', 'Polk')

TYPE\_NAME101('Donald', 'J', 'Trump')

SQL> **select c\_add from cust101;**

C\_ADD(STREET, CITY, PINCODE)

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

TYP\_ADDRESS101('White House', 'Washington', 123)

TYP\_ADDRESS101('White House', 'Washington', 123)

TYP\_ADDRESS101('White House', 'Washington', 123)

TYP\_ADDRESS101('White House', 'Washington', 123)

TYP\_ADDRESS101('White House', 'Washington', 123)

SQL> **select c\_phno from cust101;**

C\_PHNO

----------

1256468

354688

13546846

467765464

13848463

**CREATE TABLE emp101**

**(**

**emp\_no NUMERIC(5),**

**fname VARCHAR2 (10),**

**lname VARCHAR2(10),**

**salary NUMERIC(5),**

**comm NUMERIC(5)**

**);**

1 table created

INSERT INTO emp101 VALUES (&emp\_no,'&fname','&lname',&salary,&comm);

>>inserted 10 values

**select \* from emp101;**

EMP\_NO FNAME LNAME SALARY COMM

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

101 Donald Trump 50000 5600

102 Jason Mamoa 45000 560

103 John Adams 52000 21

104 George Washington 89000 2500

105 James Monroe 80000 5600

106 Andrew Jackson 14000 560

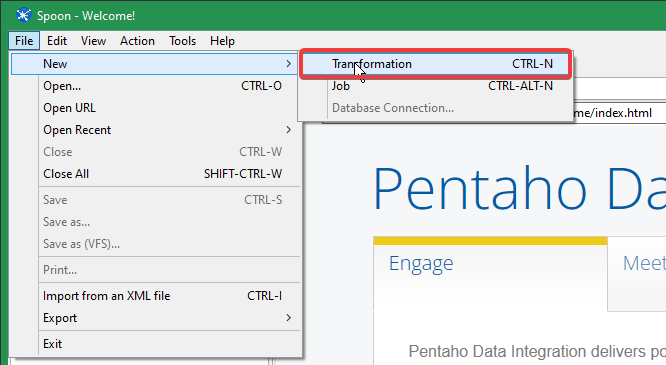
107 William Harrison 59000 650

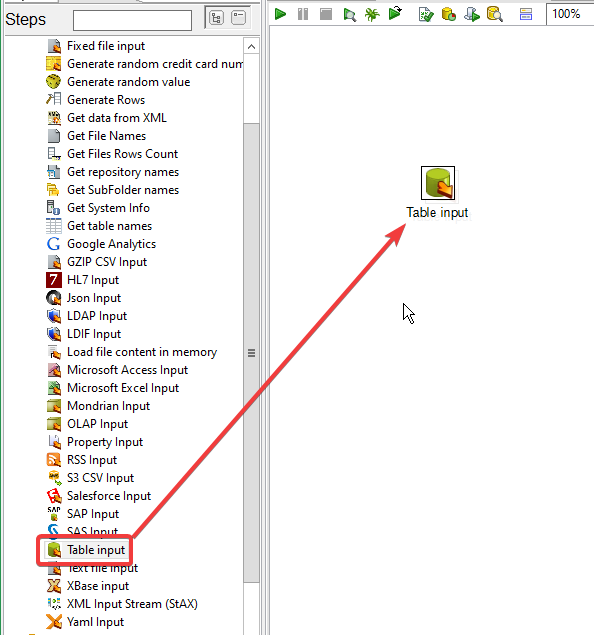
108 John Tyler 75000 450

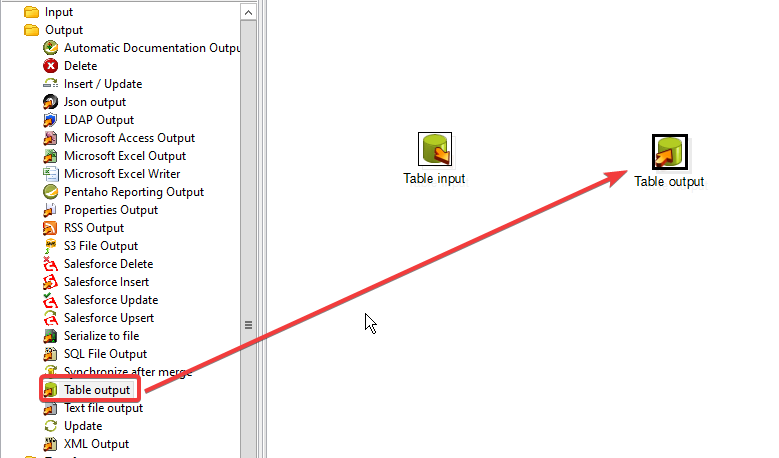
109 James Polk 56222 226

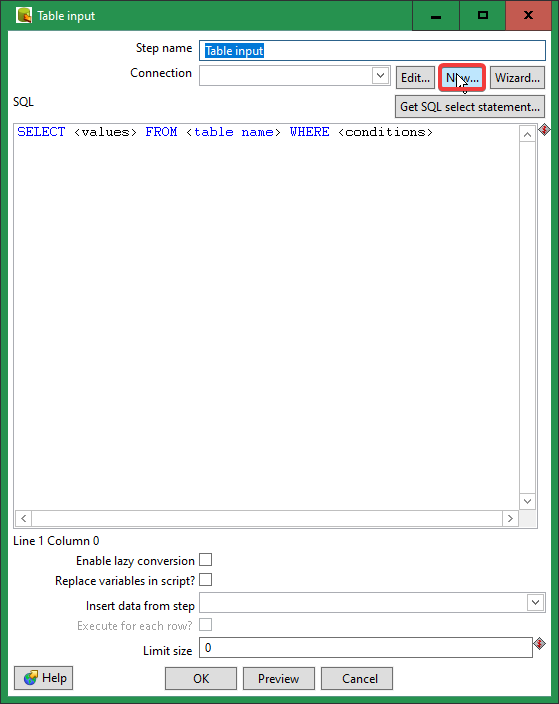
110 James Buchanan 85444 444

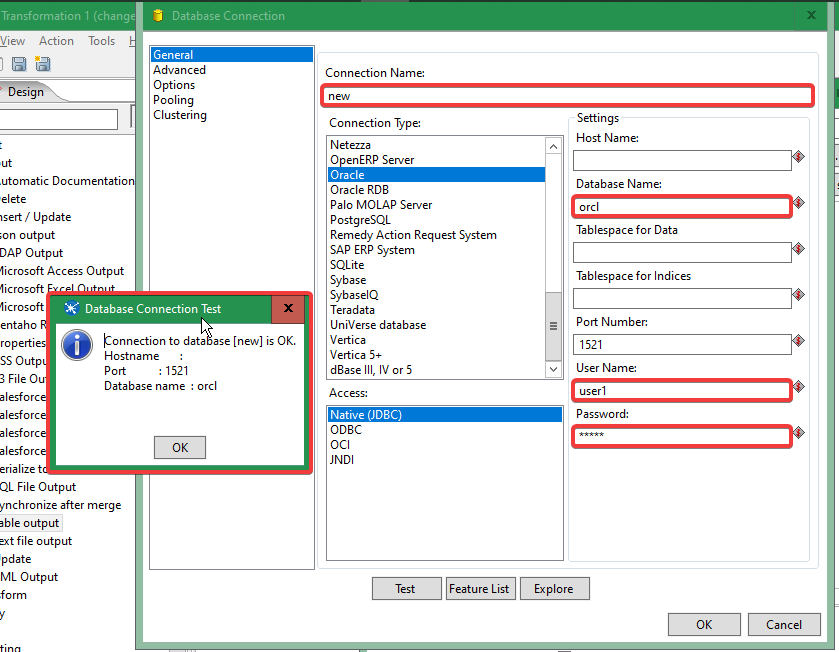
10 rows selected.

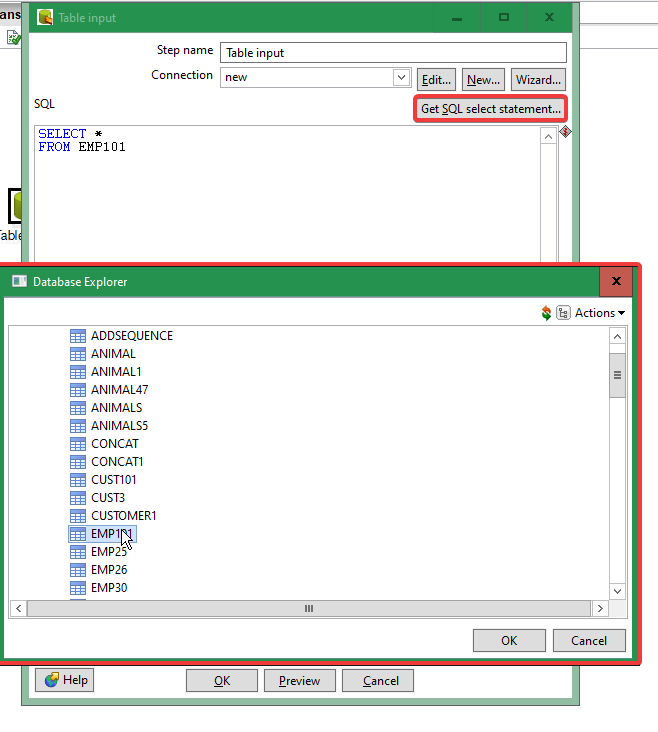




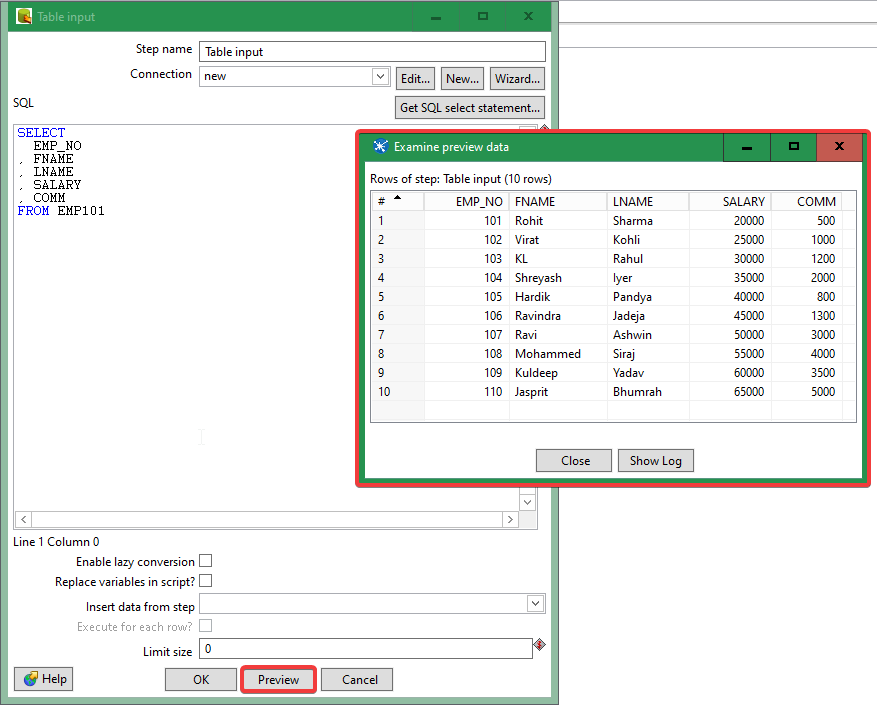




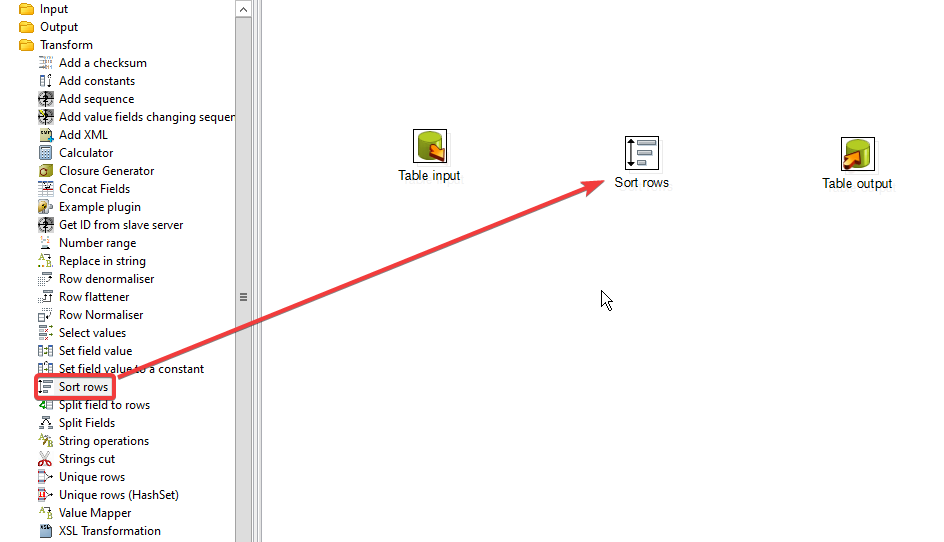


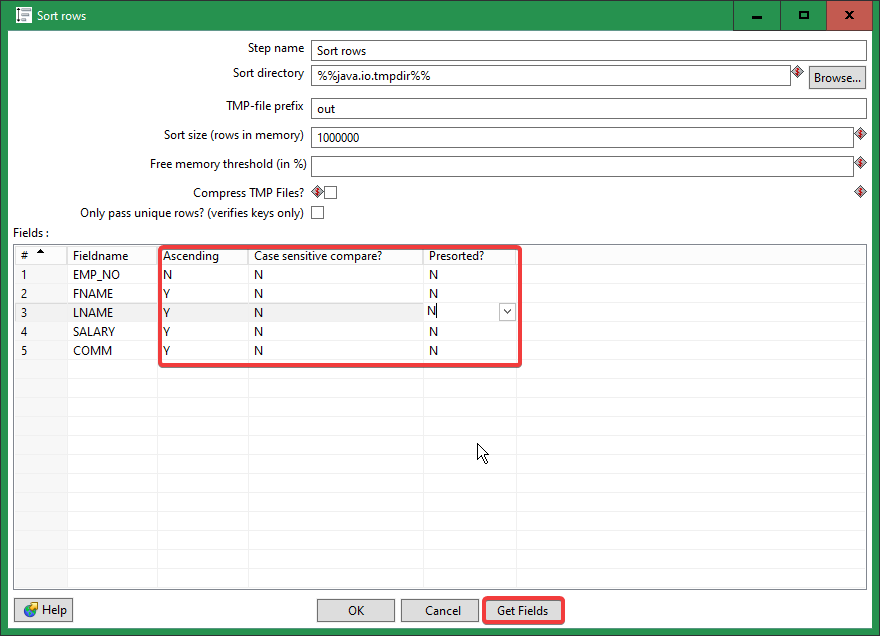


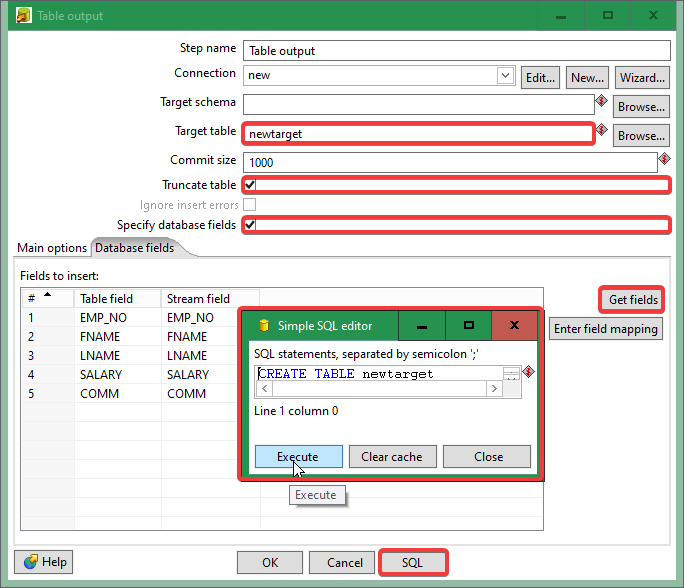
In the new pop-up click yes for including fields.



FOR SORT ROWS







SQL> select \* from newtarget;

EMP\_NO FNAME LNAME SALARY COMM

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

110 Jasprit Bhumrah 65000 5000

109 Kuldeep Yadav 60000 3500

108 Mohammed Siraj 55000 4000

107 Ravi Ashwin 50000 3000

106 Ravindra Jadeja 45000 1300

105 Hardik Pandya 40000 800

104 Shreyash Iyer 35000 2000

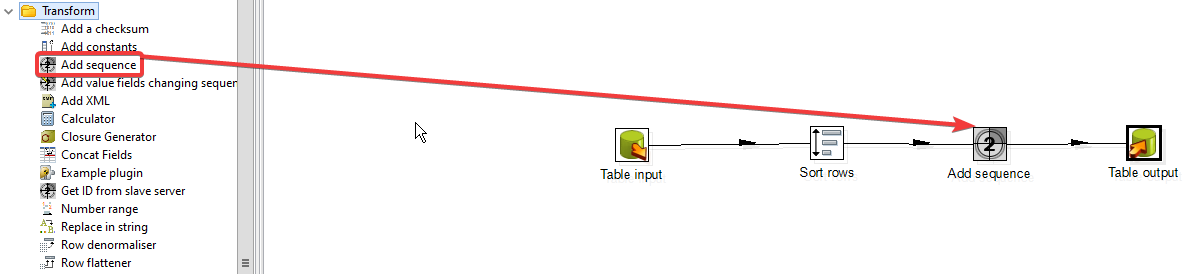
103 KL Rahul 30000 1200

102 Virat Kohli 25000 1000

101 Rohit Sharma 20000 500

10 rows selected.

Adding sequence:



Sequence setting:

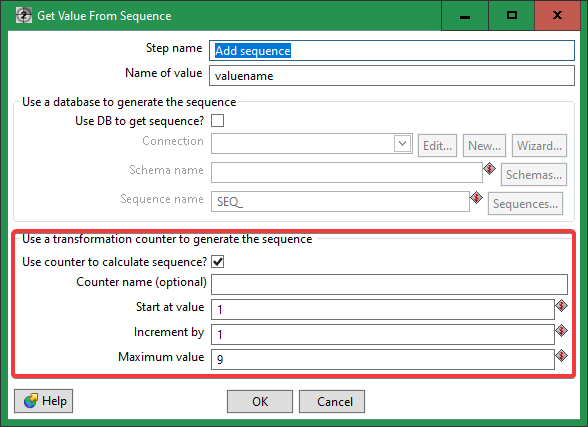
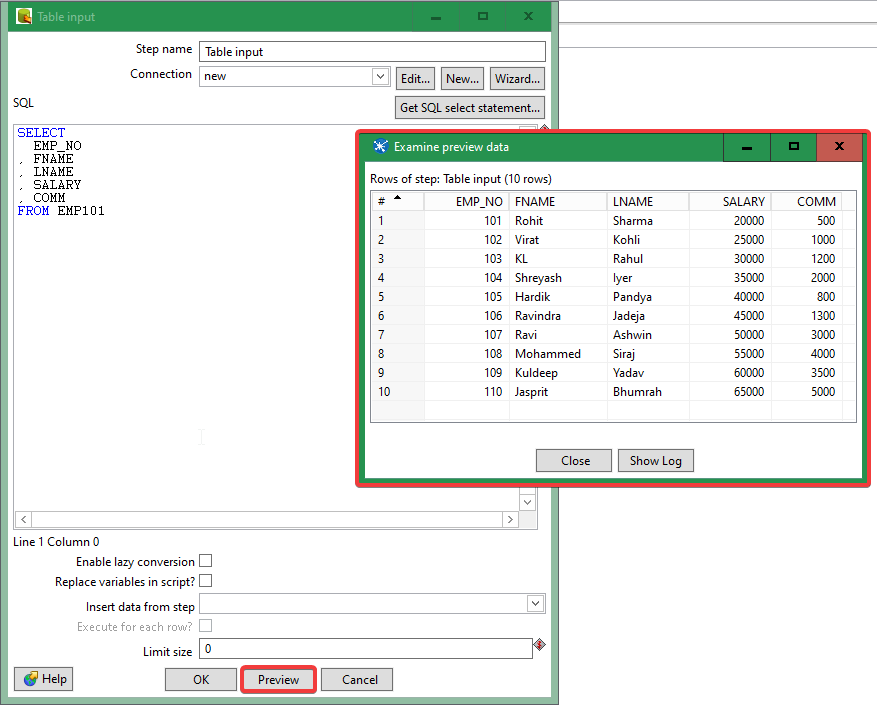


Table Input setting:



Sort rows setting:

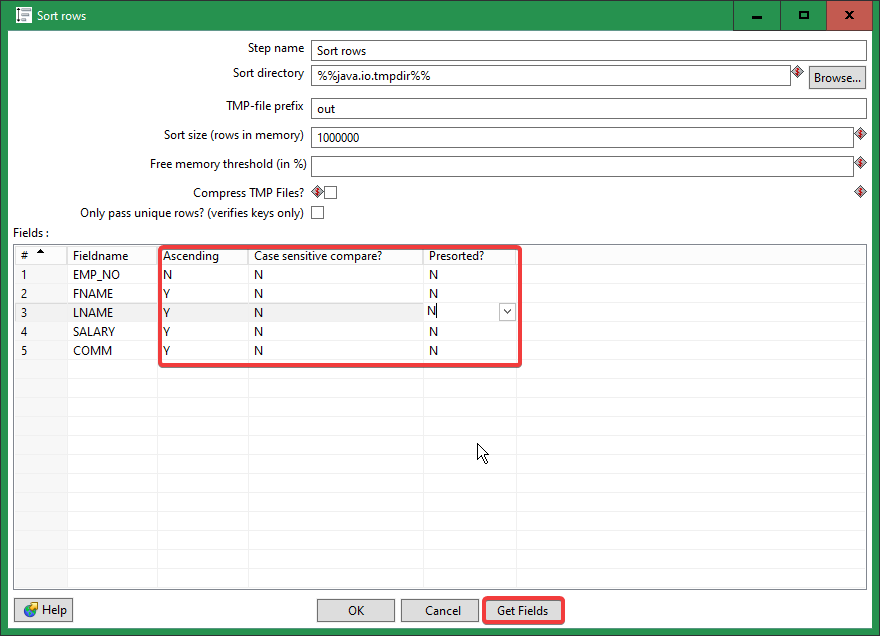
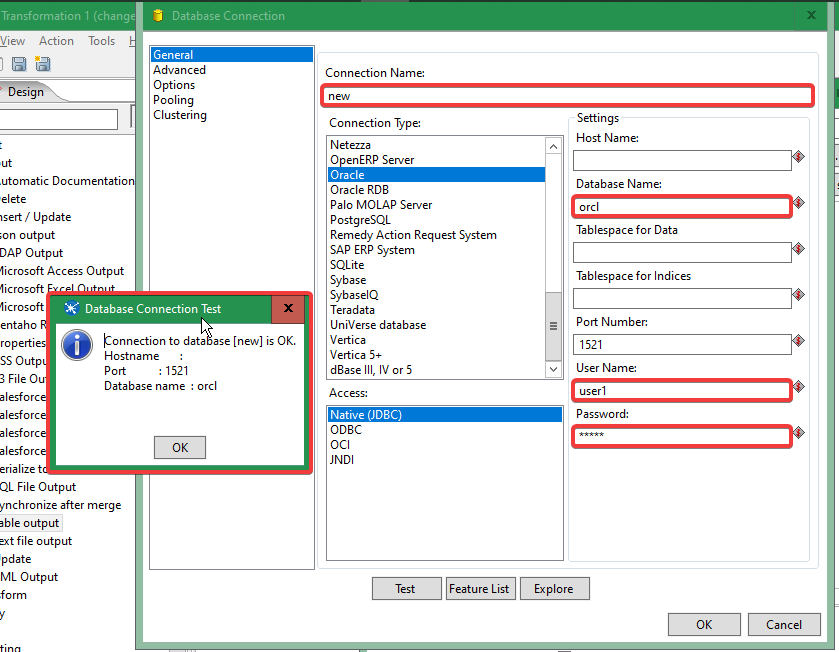
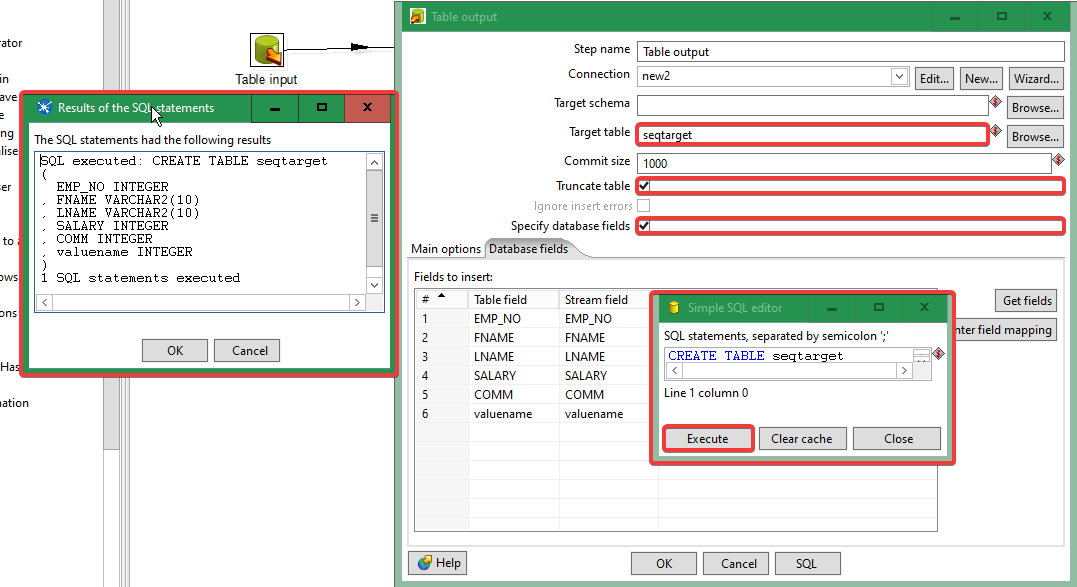
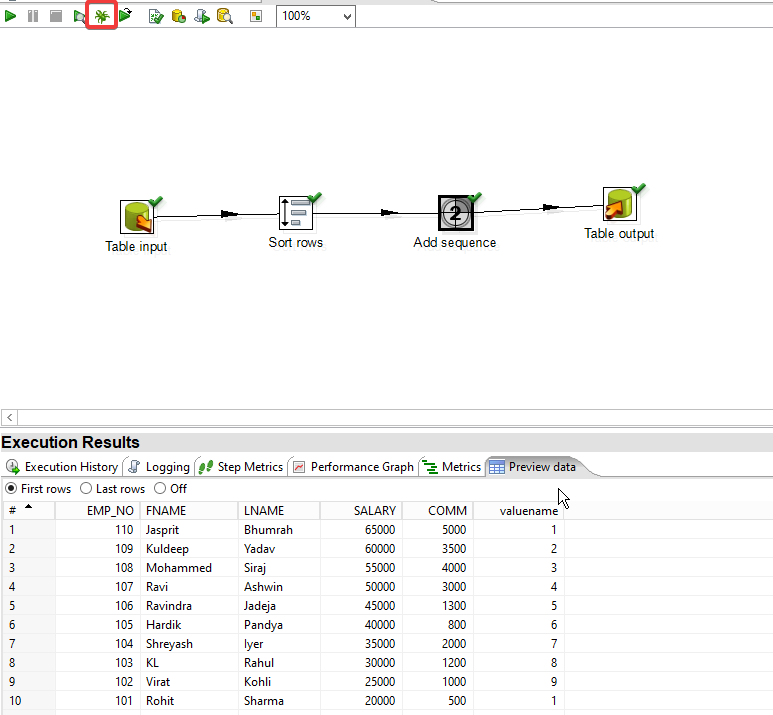


Table Output setting:





O/P:



Calculator Operation:

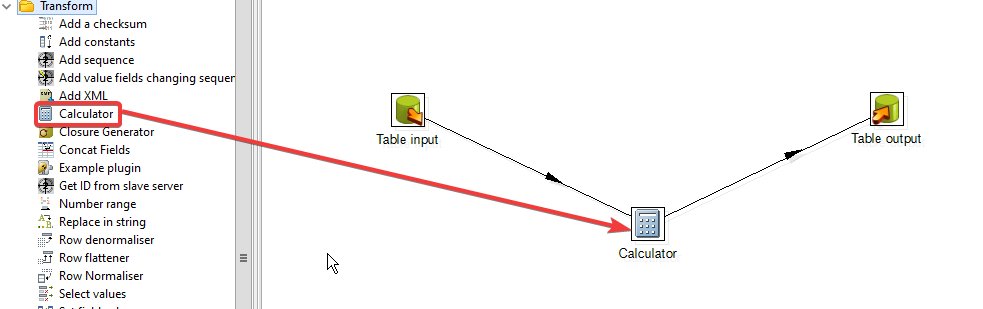


Table Input setting:

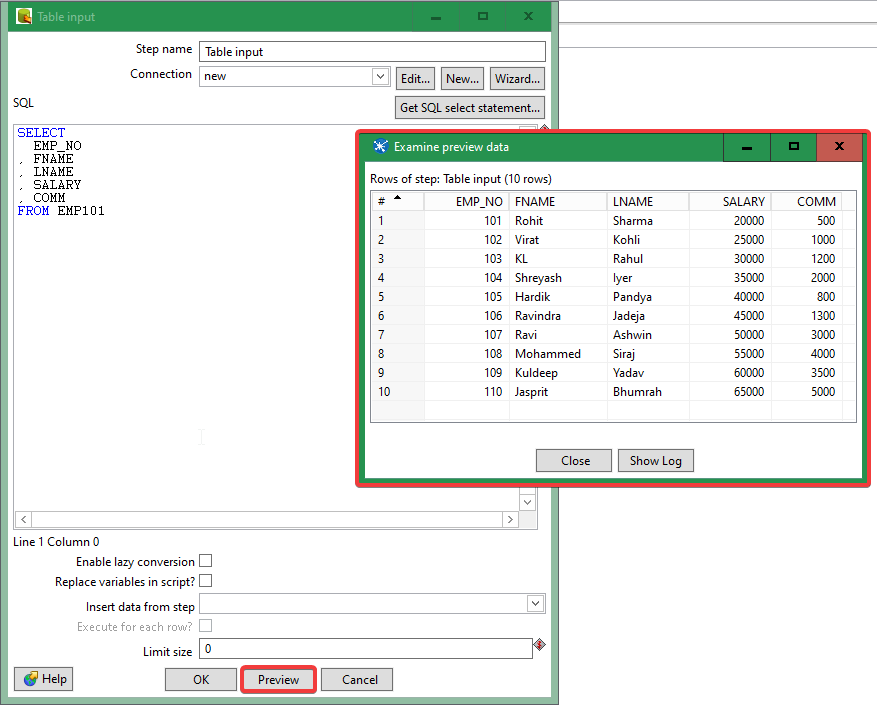
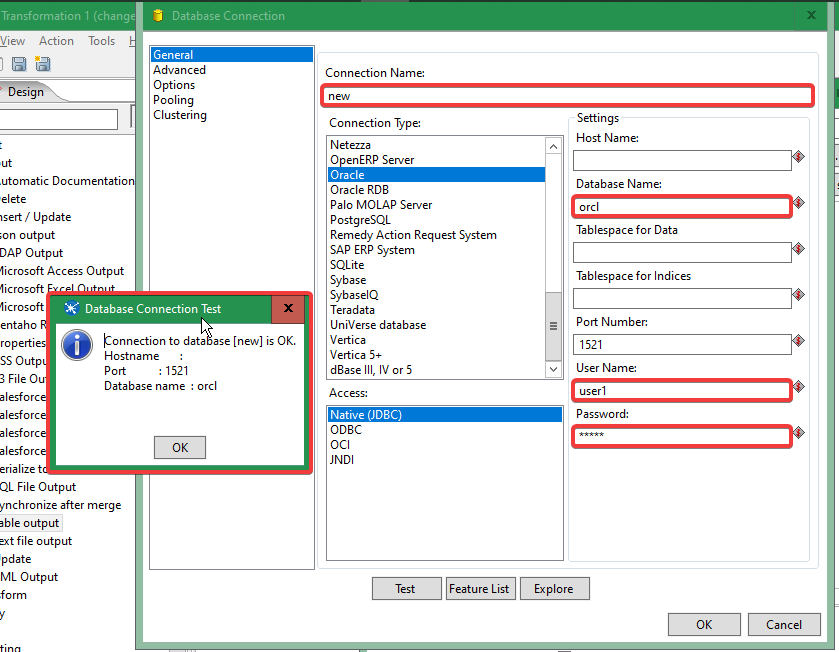
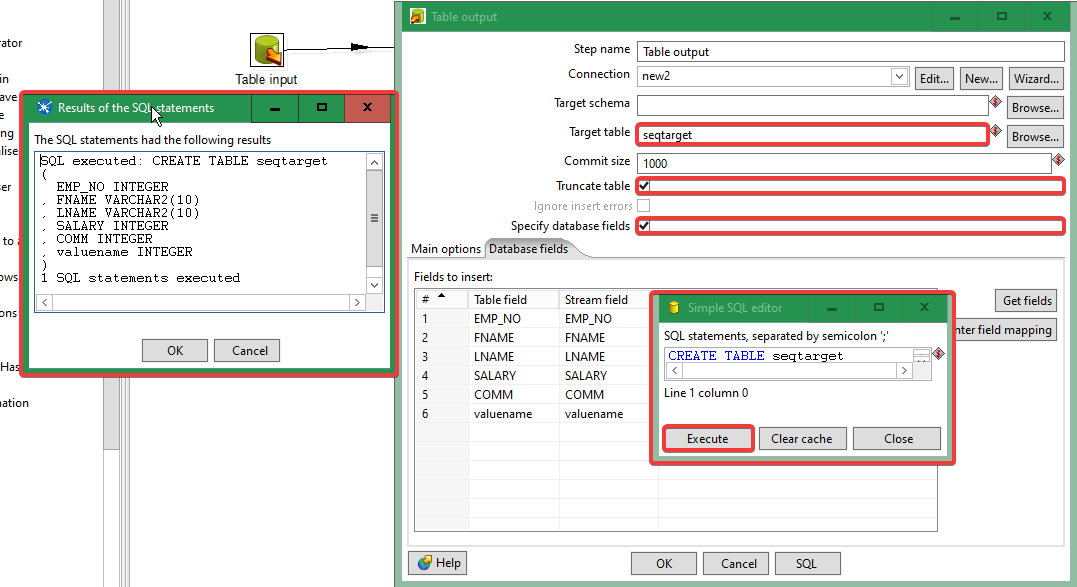
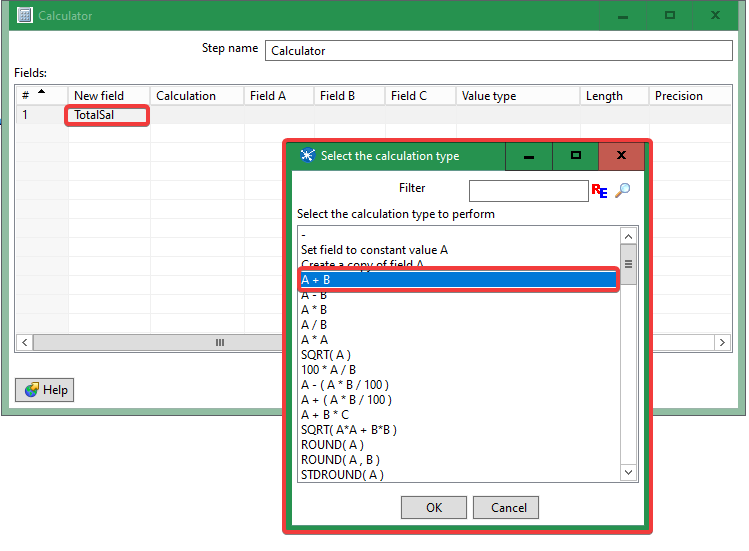


Table OutPut setting:





Calculator Setting:



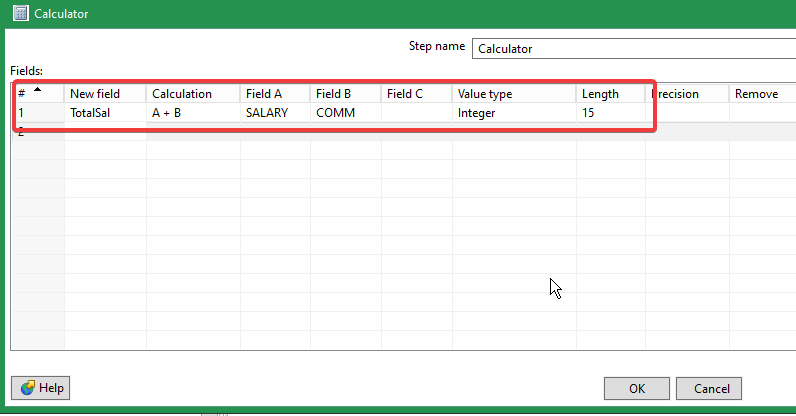
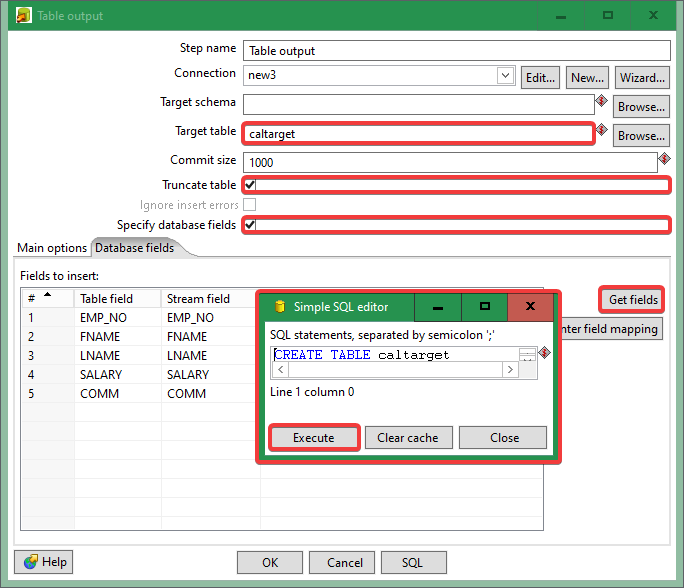
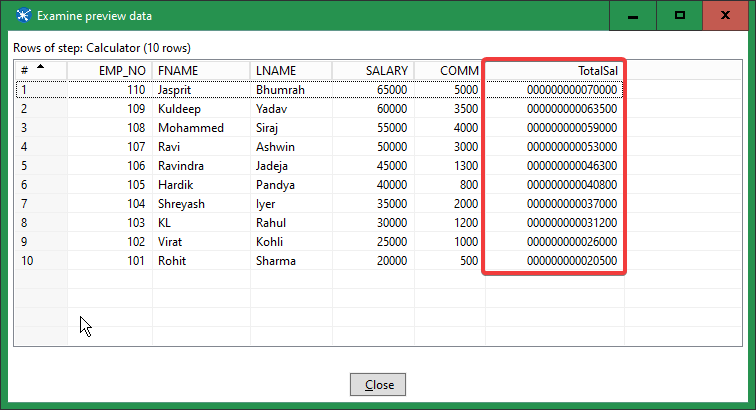


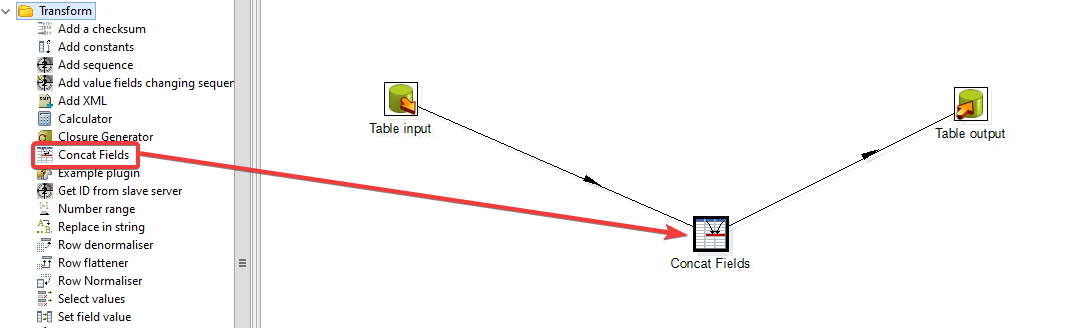
Table Output setting:

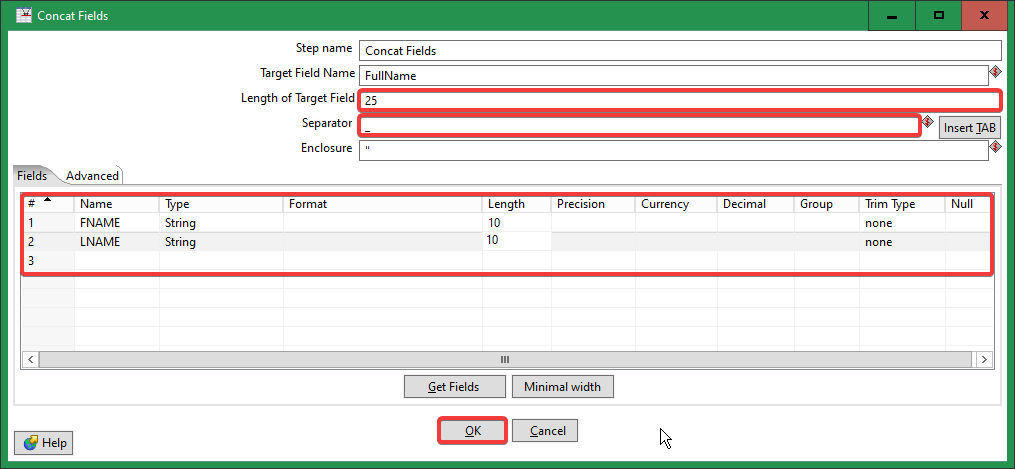


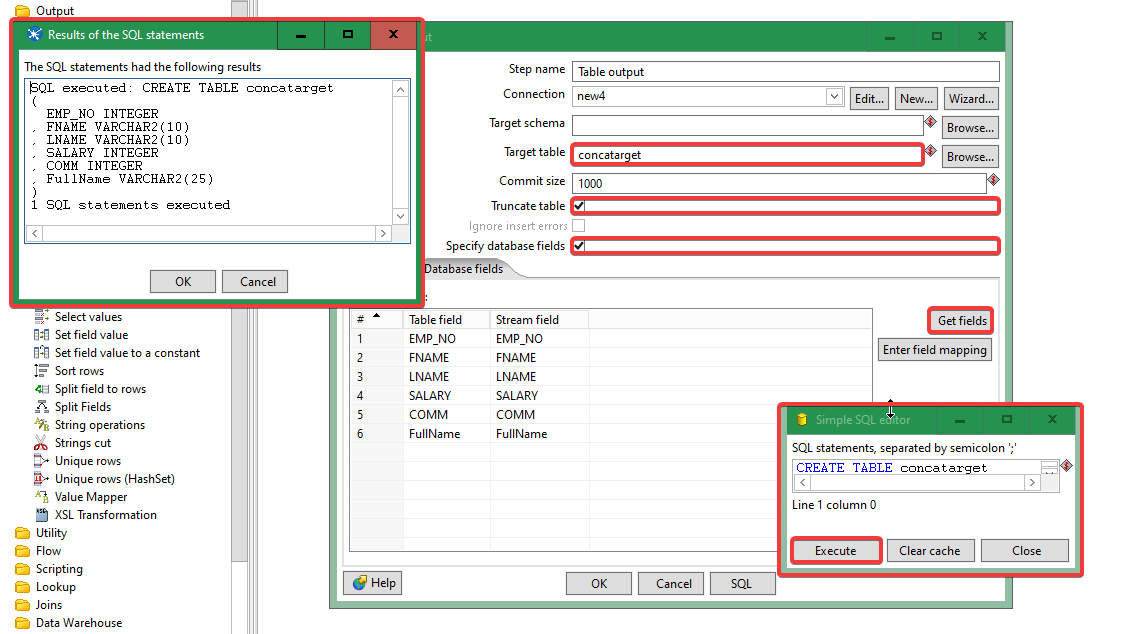
O/P:



Concatenation:







Splitting:

