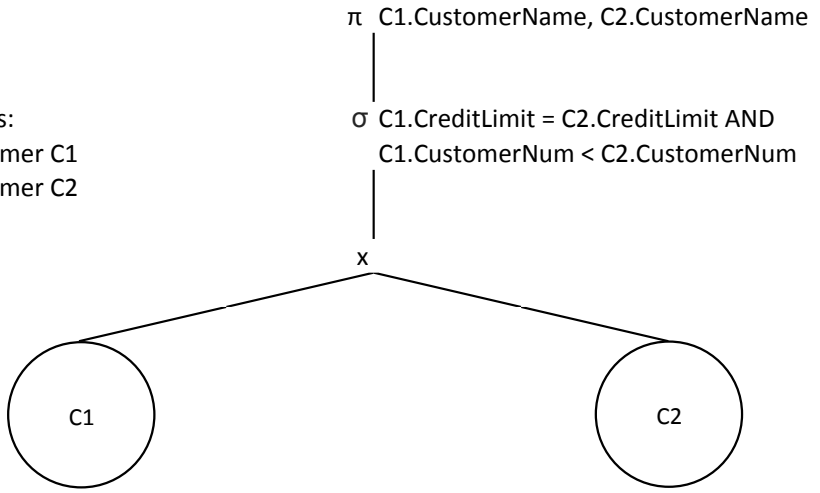
Name: Manuel Cespedes
Panther-ID: 1730088
Course: COP 4722
Assignment#: 1
Due: Tues, Sep 23, 2014

I hereby certify that this work is my own and none of it is the work of any other person.

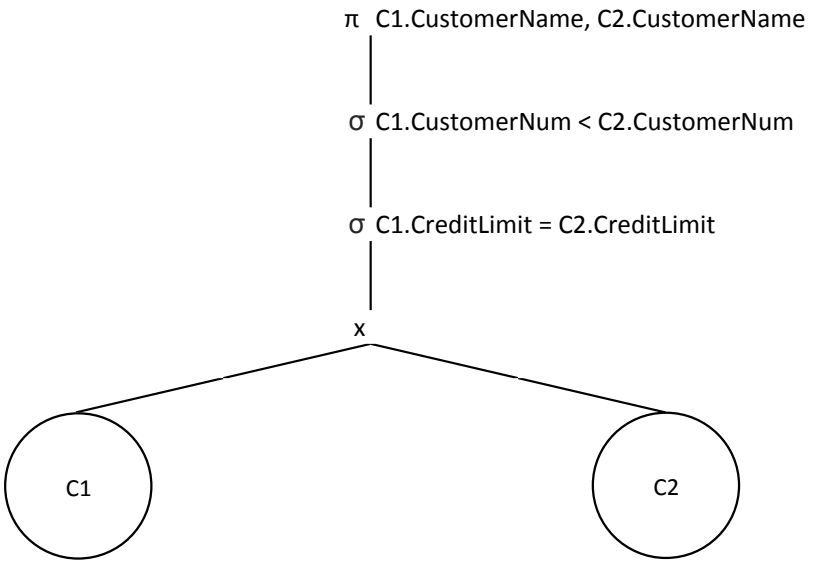
Signature: Manuel G. Cespedes

Q1) Initial query tree:



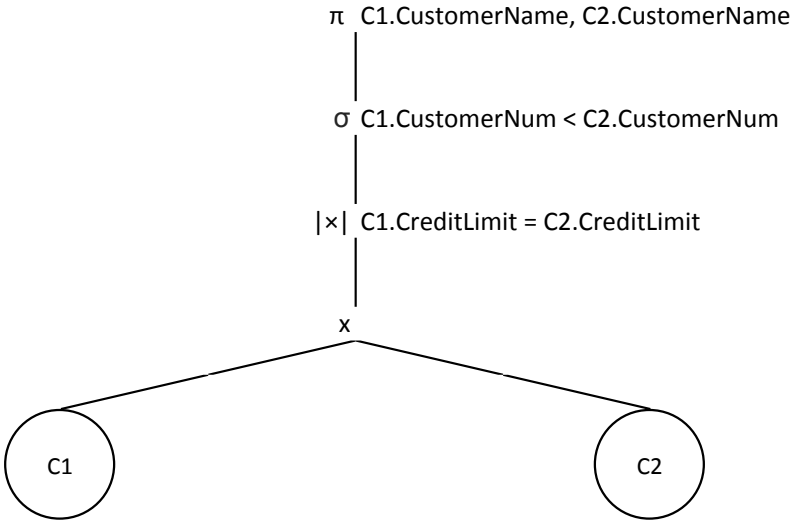
Tables:
Customer C1
Customer C2

Step 1:

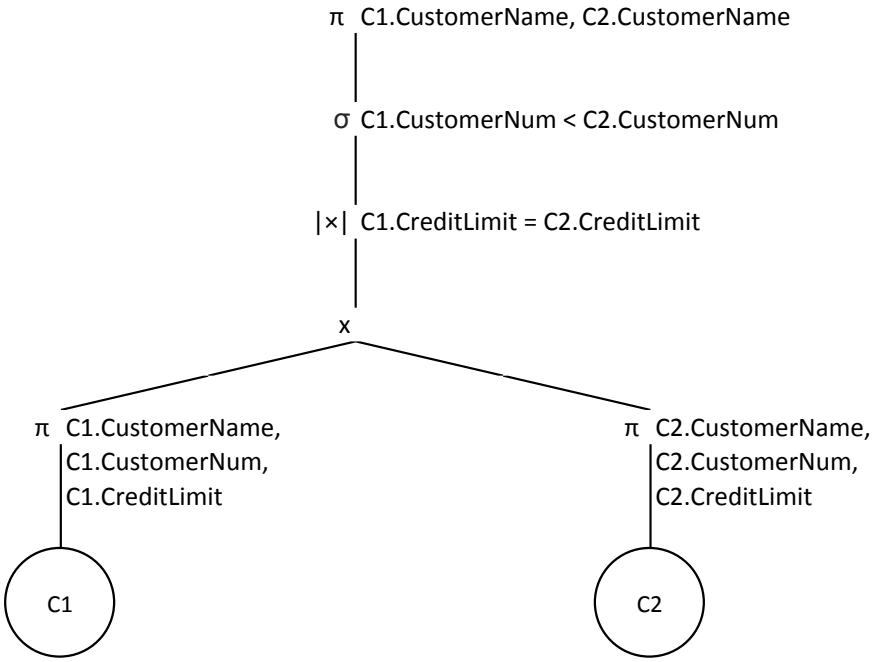


No Step 2

Step 3:



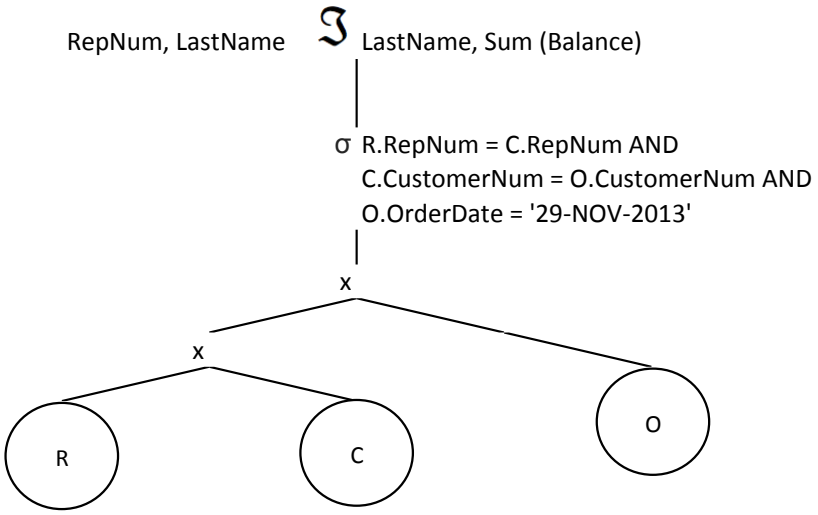
Step 4:



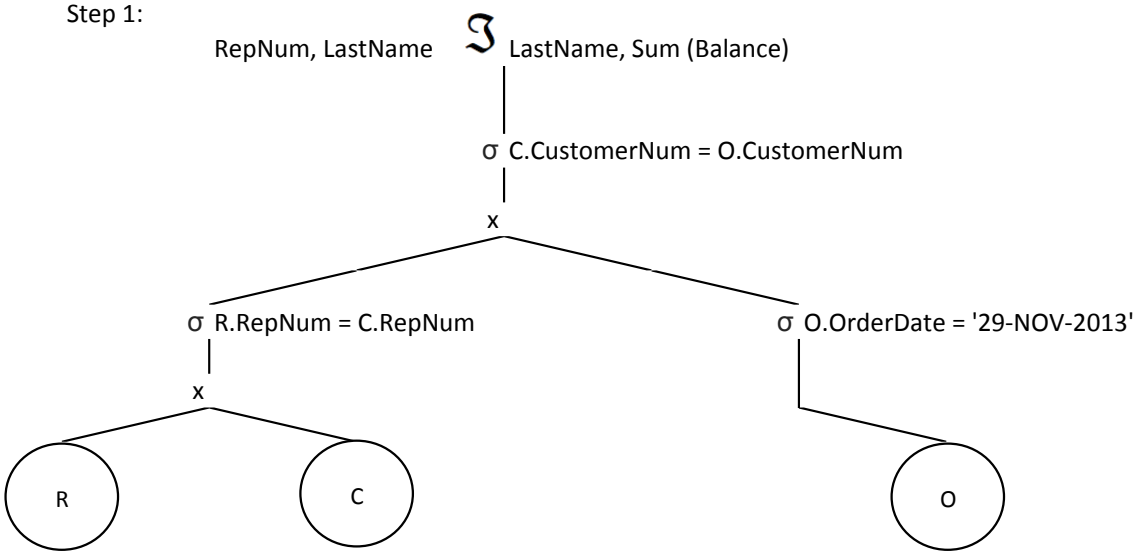
Optmized query:

```
SELECT    C1.CustomerName, C2.CustomerName
FROM      ( SELECT CustomerName, CreditLimit, CustomerNum
          FROM Customer) as C1,
          ( SELECT CustomerName, CreditLimit, CustomerNum
          FROM Customer) as C2
WHERE     C1.CreditLimit = C2.CreditLimit AND
          C1.CustomerNum < C2.CustomerNum
```

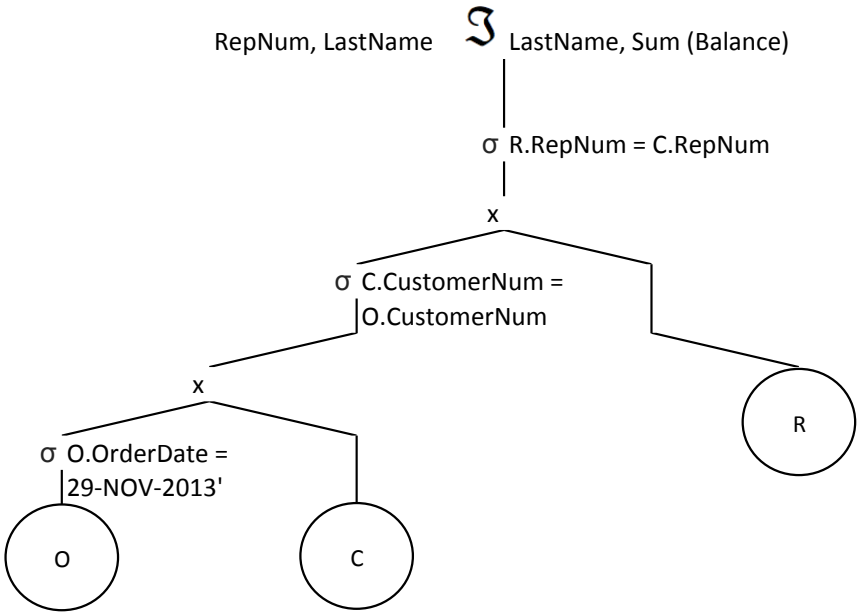
Q2) Initial query tree:



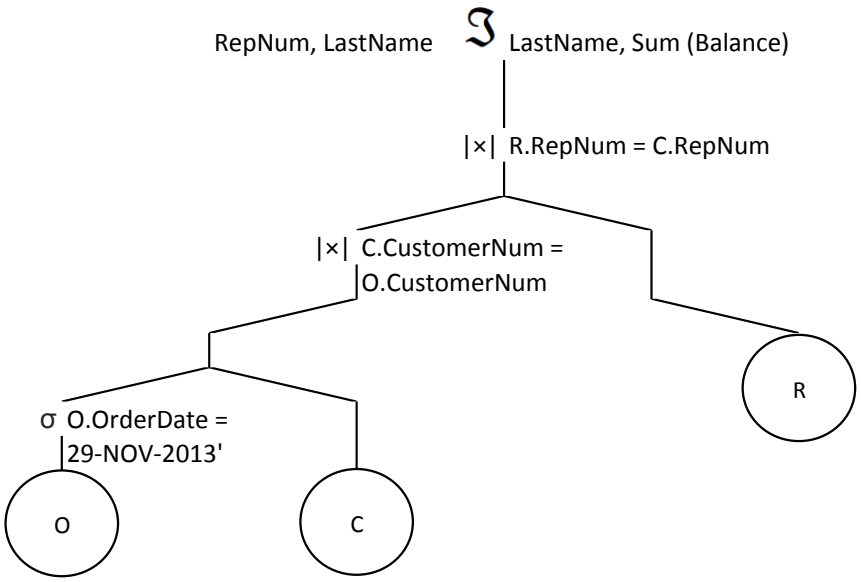
Step 1:



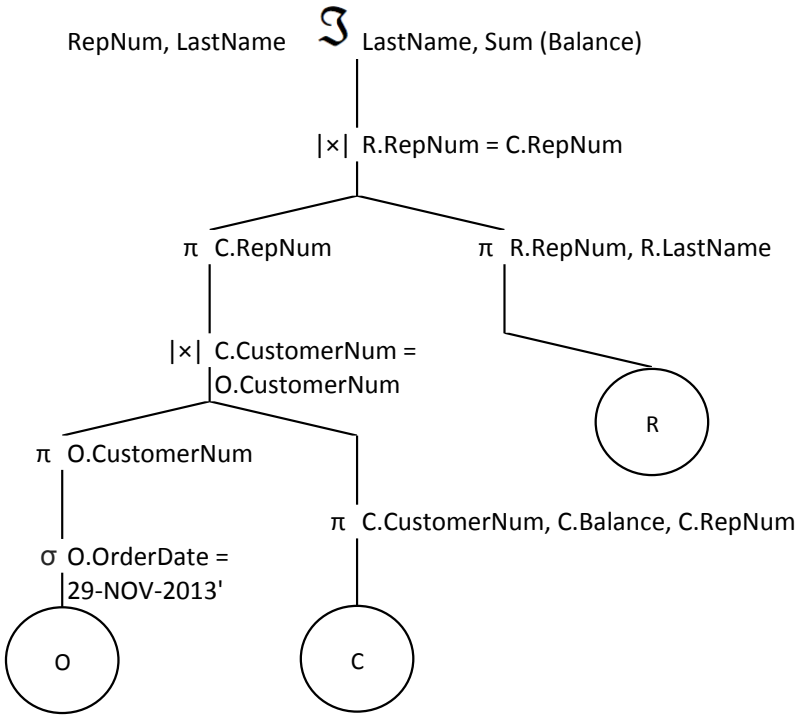
Step 2:


RepNum, LastName
ℑ
LastName, Sum (Balance)
σ R.RepNum = C.RepNum
x
σ C.CustomerNum =
O.CustomerNum
x
R
σ O.OrderDate =
29-NOV-2013'
O
C

Step 3:


RepNum, LastName
ℑ
LastName, Sum (Balance)
|×| R.RepNum = C.RepNum
|×| C.CustomerNum =
O.CustomerNum
R
σ O.OrderDate =
29-NOV-2013'
O
C

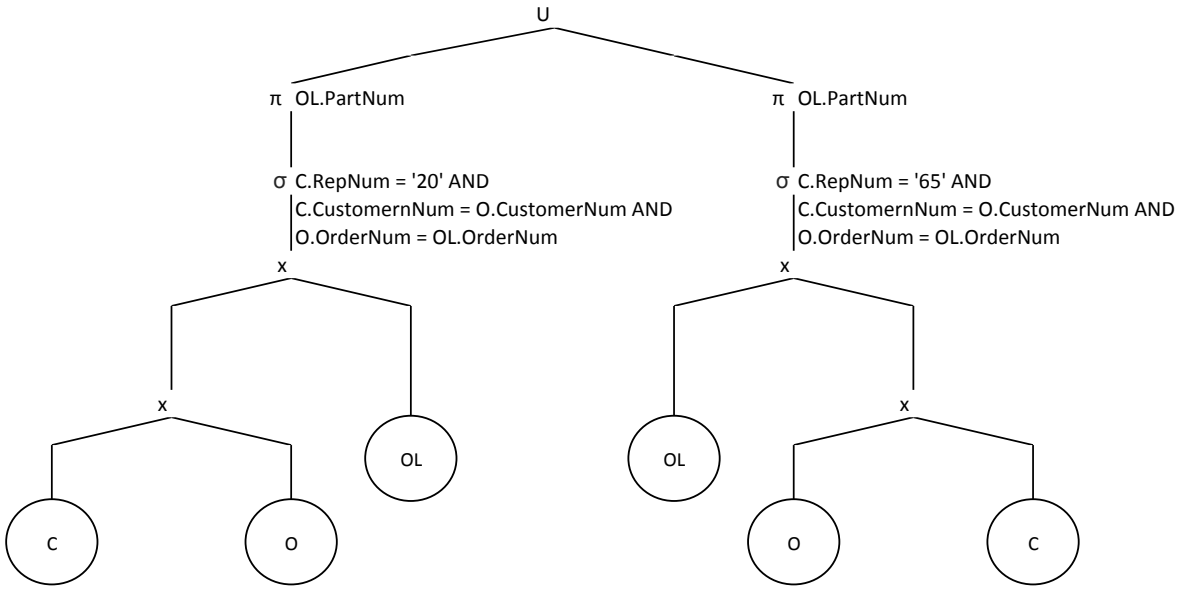
Step 4:



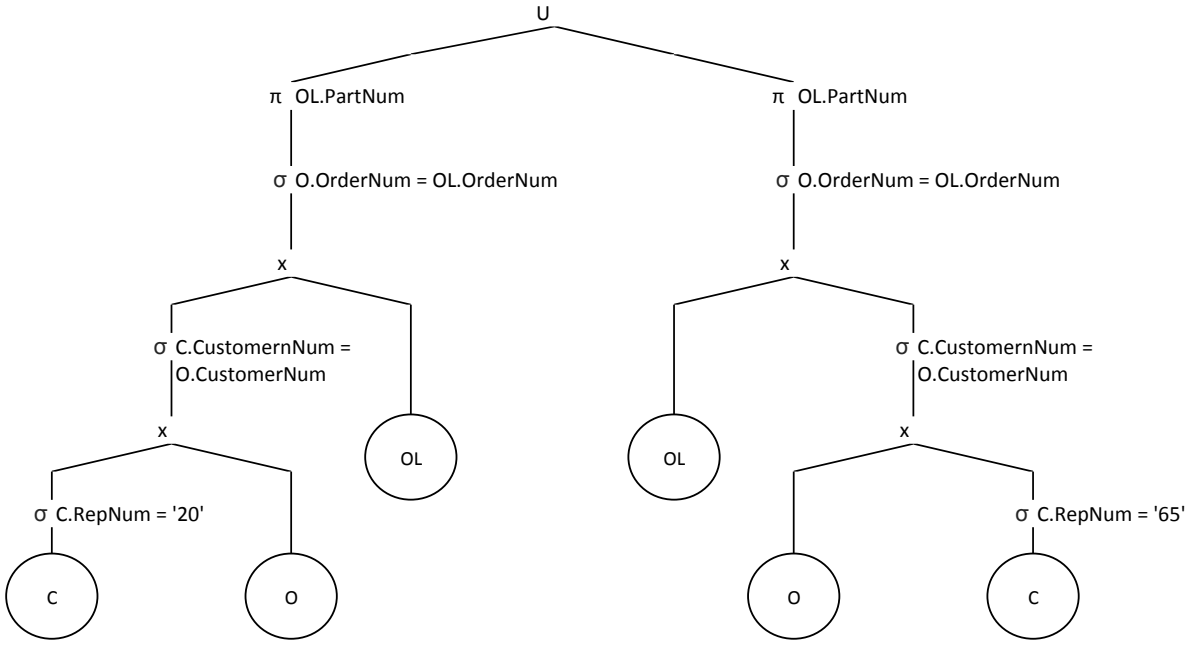
Optmized query:

```
SELECT      R.LastName, Sum (C.Balance)
FROM        Rep R, Customer C
WHERE       R.RepNum IN
            ( SELECT      C.RepNum
            FROM          Customer C
            WHERE         C.CustomerNum IN
                          ( SELECT      O.CustomerNum
                          FROM          Orders O
                          WHERE         O.OrderDate = '29-NOV-2013'))
GROUP BY    R.RepNum, R.LastName
```

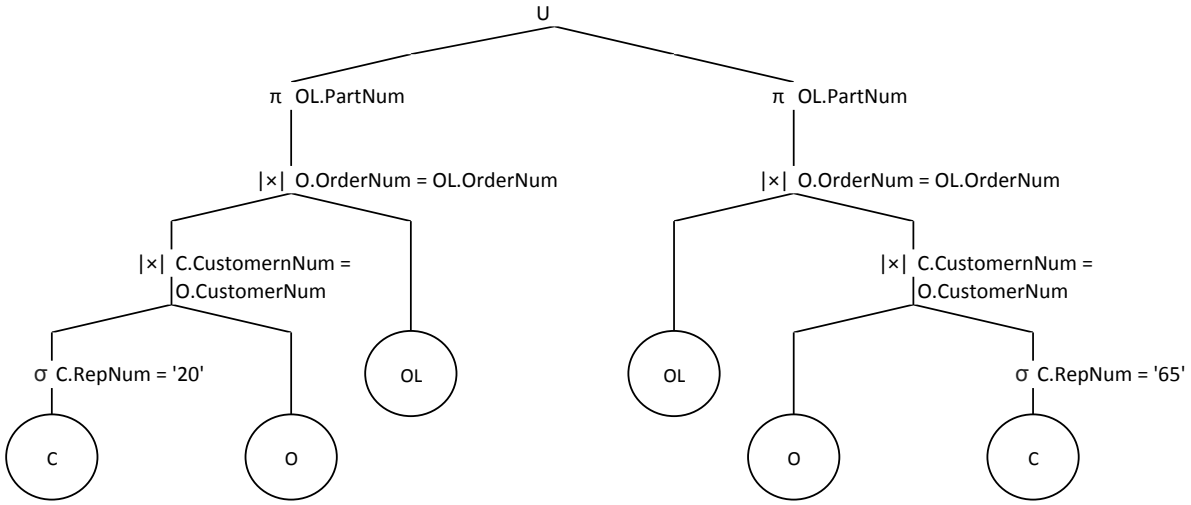
Q3) Initial query tree:



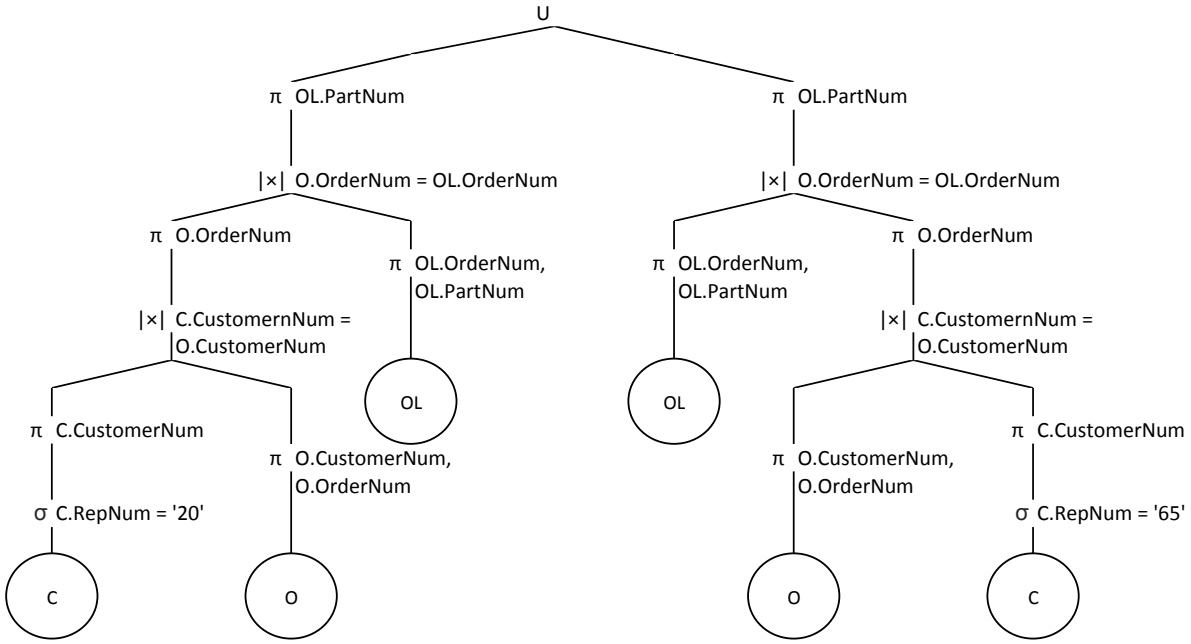
Step 1:



Step 2 / Step 3:



Step 4:

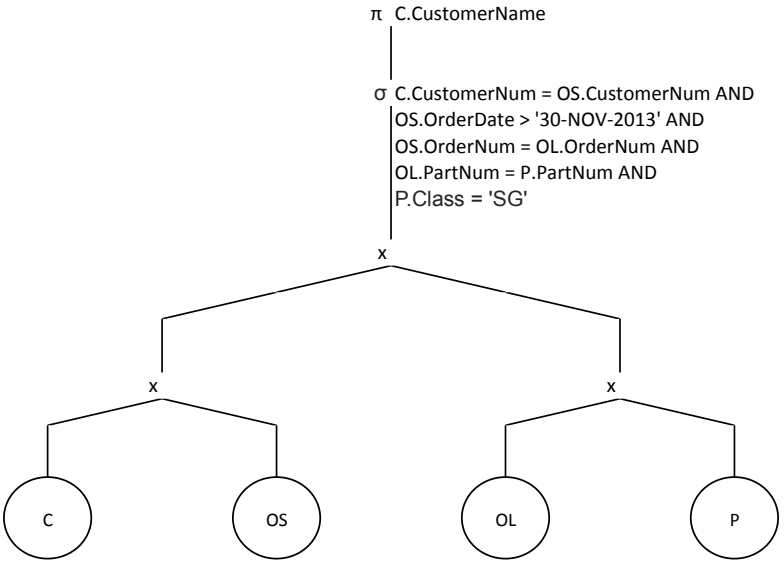


Optmized query:

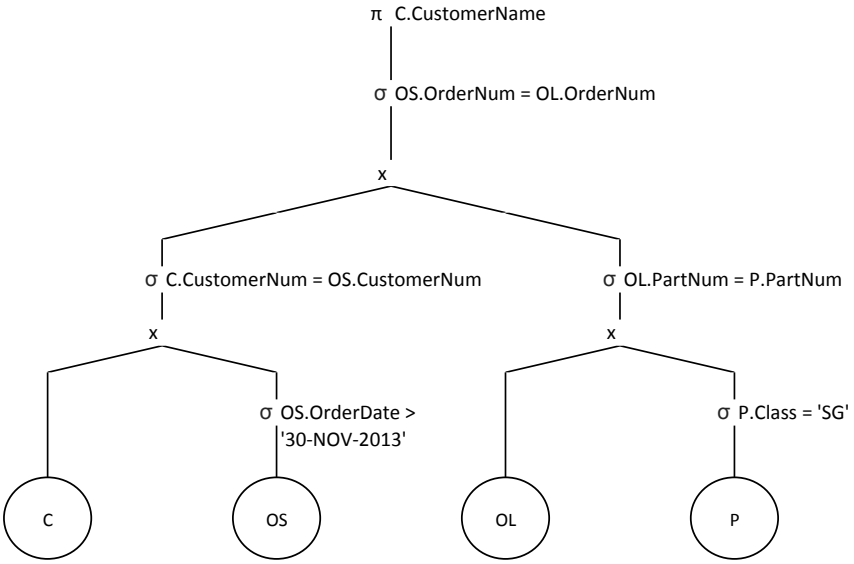
```
SELECT      OL.PartNum
FROM        OrderLine OL
WHERE       OL.OrderNum IN
            ( SELECT      O.OrderNum
            FROM          Orders O
            WHERE         O.CustomerNum IN
                          ( SELECT      C.CustomerNum
                          FROM          Customer C
                          WHERE         C.RepNum = '20'))
UNION

SELECT      OL.PartNum
FROM        OrderLine OL
WHERE       OL.OrderNum IN
            ( SELECT      O.OrderNum
            FROM          Orders O
            WHERE         O.CustomerNum IN
                          ( SELECT      C.CustomerNum
                          FROM          Customer C
                          WHERE         C.RepNum = '65'))
```

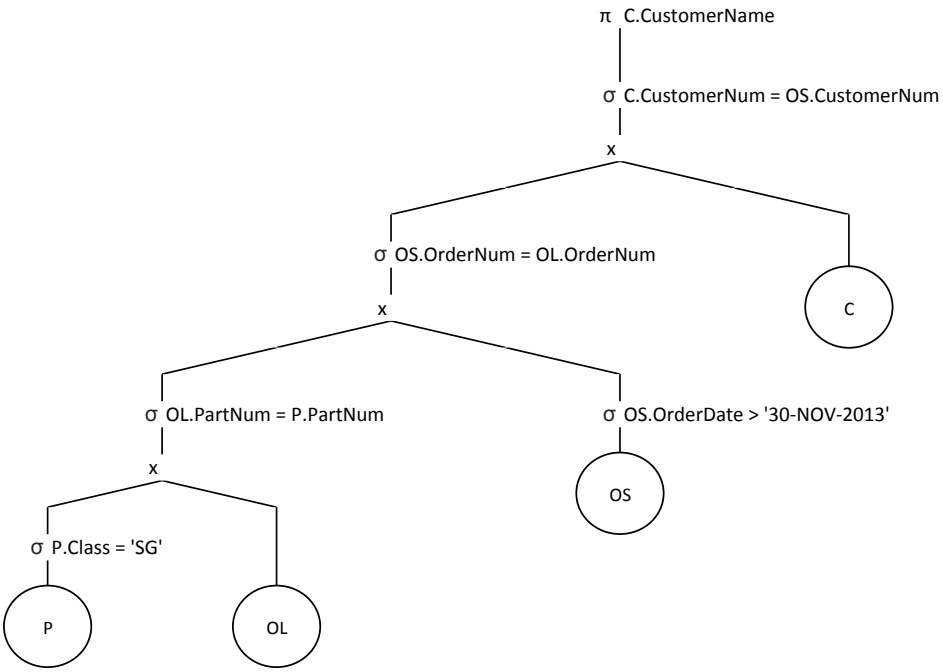
Q4) Initial query tree:



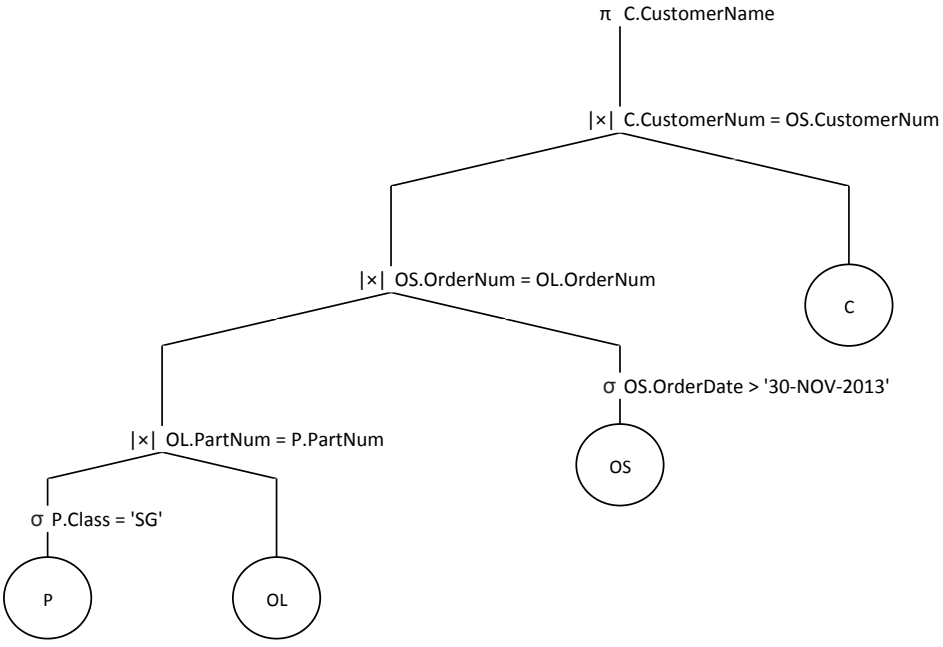
Step 1:



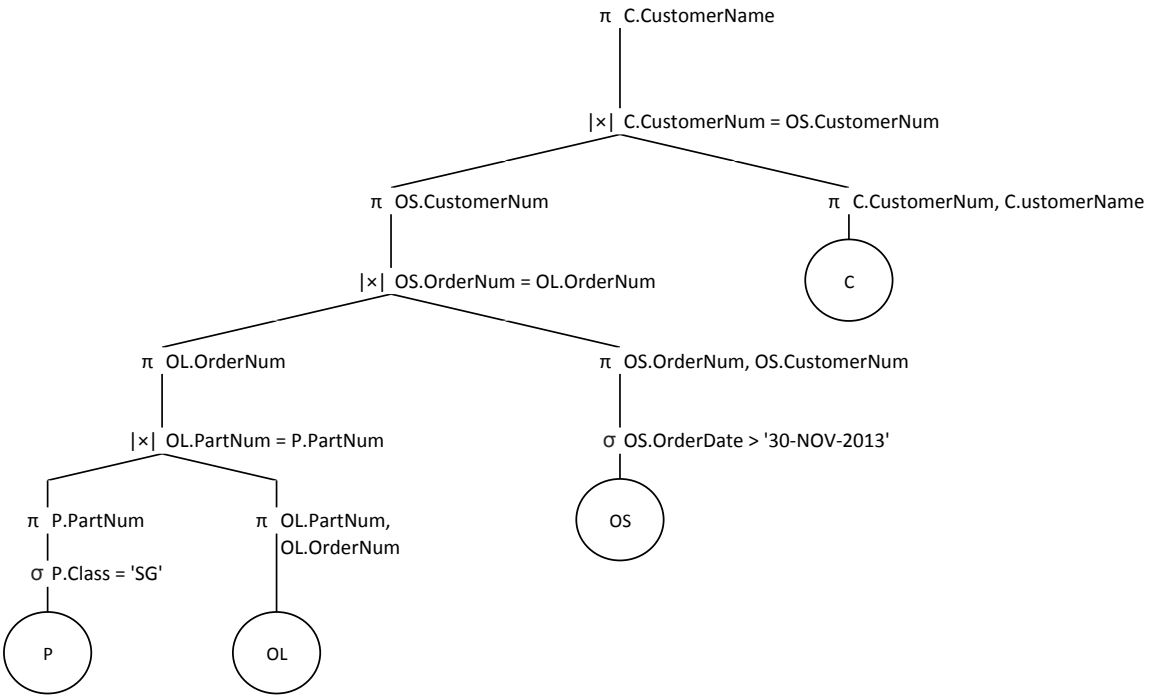
Step 2:


π C.CustomerName
σ C.CustomerNum = OS.CustomerNum
x
σ OS.OrderNum = OL.OrderNum
C
x
σ OL.PartNum = P.PartNum
σ OS.OrderDate > '30-NOV-2013'
x
OS
σ P.Class = 'SG'
P
OL

Step 3:


π C.CustomerName
|×| C.CustomerNum = OS.CustomerNum
|×| OS.OrderNum = OL.OrderNum
C
σ OS.OrderDate > '30-NOV-2013'
|×| OL.PartNum = P.PartNum
OS
σ P.Class = 'SG'
P
OL

Step 4:



Optmized query:

```
SELECT    C.CustomerName
FROM      Customer C
WHERE     C.CustomerNum IN
          ( SELECT    OS.CustomerNum
          FROM        Orders OS
          WHERE       OS.OrderDate > '30-NOV-2013' AND OS.OrderNum IN
                      ( SELECT    OL.OrderNum
                      FROM        OrderLine OL
                      WHERE       OL.PartNum IN
                                  ( SELECT    P.PartNum
                                  FROM        Part P
                                  WHERE       P.Class = 'AG')));
```

```
// 19_3a.txt Page 1
// Modified Figure 19.3 (a)
// ========================
// Implementation of the full-outer join between tables R and S with (R.A = S.B)
// Extension of sort-merge join
// sort the tuples in R on attribute A; // assume R has n tuples (records)
// sort the tuples in S on attribute B; // assume S has m tuples (records)
// i = 1; // initialize the record pointer of table R
// j = 1; // initialize the record pointer of table S

while ((i <= n) && (j <= m))
{
        if (R[i].A > S[j].B)
        {
                print(S[j])
                S[j].B = null // assign the table cell as NULL
                j++; // advance the record pointer of S;
        }
        elseif (R[i].A < S[j].B )
        {
                print(S[i])
                R[j].A = null // assign the table cell as NULL
                i++; // advance the record pointer of R
        }
        else
        { // R[i].A == S[j].B, so we output all matched pairs of tuples
                p = i; // p is the auxillary record pointer of table R
                while ((p <= n) && (R[p].A == S[j].B))
                {
                        q = j; // q is the auxillary record pointer of table S
                        while ((q <= m) && (R[p].A == S[q].B))
                        {
                                output the combined tuple <R[p],S[q]> to T; //T is the
result table
                                q++;
                        }
                        p++;
                }
                i = p; // update the primary record pointer of table R
                j = q; // update the primary record pointer of table S
        }
}
```