Database Optimization: Practicing

ADVANCED DATA BASE

Business Intelligence Group

Matteo Golfarelli

Exercise 1

Determine the optimized execution tree with respect to the heuristic criteria presented in class

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    ORDERS,CUSTOMER,NATION,REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```

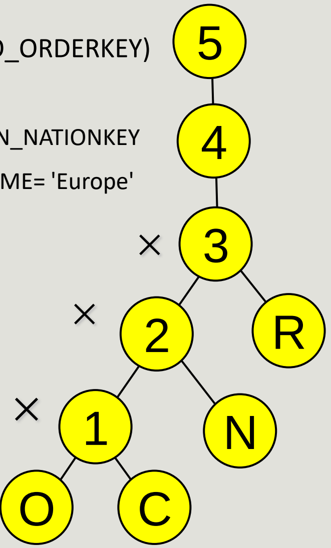
Exercise 1

Determine the optimized execution tree with respect to the heuristic criteria presented in class

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    ORDERS,CUSTOMER,NATION,REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```

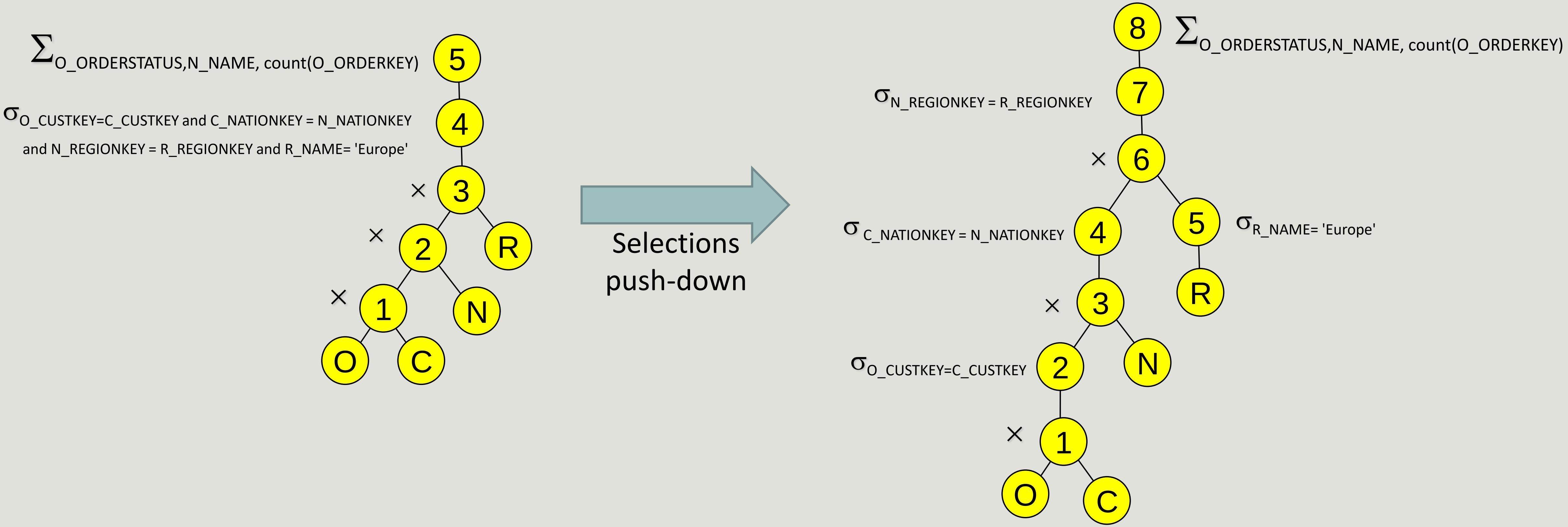
$\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$

$\sigma_{\text{O\_CUSTKEY=C\_CUSTKEY and C\_NATIONKEY = N\_NATIONKEY and N\_REGIONKEY = R\_REGIONKEY and R\_NAME= 'Europe'}}$



Exercise 1

Determine the optimized execution tree with respect to the heuristic criteria presented in class



$\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$

$\sigma_{\text{O\_CUSTKEY=C\_CUSTKEY and C\_NATIONKEY = N\_NATIONKEY and N\_REGIONKEY = R\_REGIONKEY and R\_NAME= 'Europe'}}$

Selections push-down

$\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$

$\sigma_{\text{N\_REGIONKEY = R\_REGIONKEY}}$

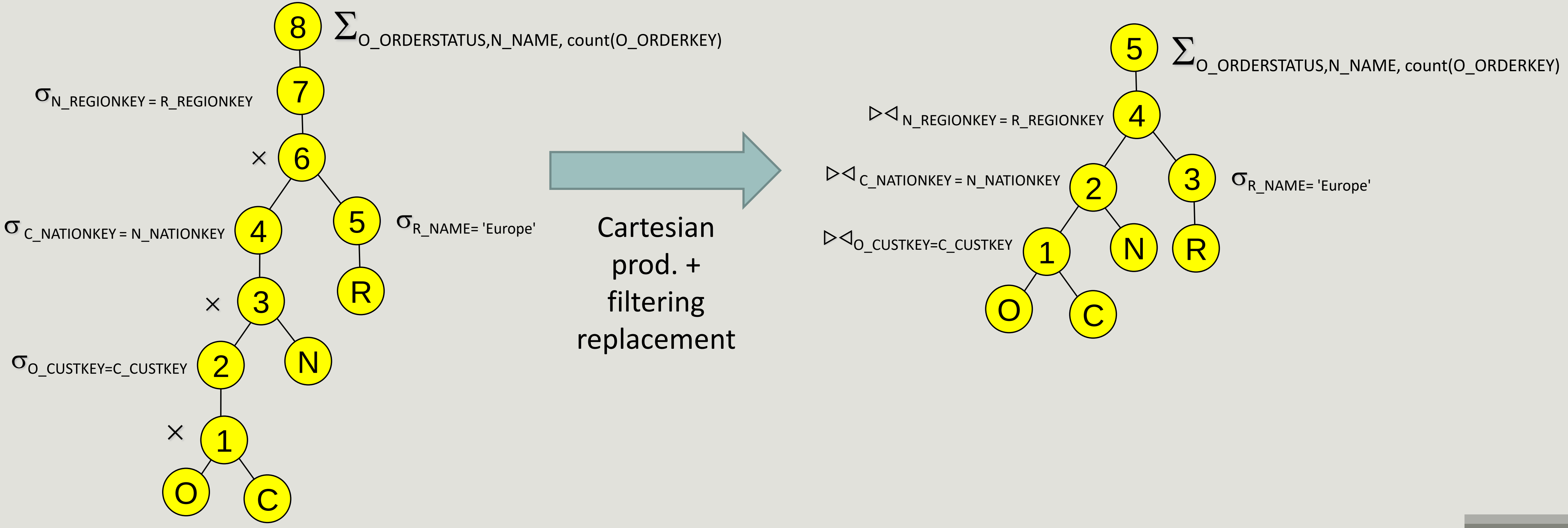
$\sigma_{\text{C\_NATIONKEY = N\_NATIONKEY}}$

$\sigma_{\text{R\_NAME= 'Europe'}}$

$\sigma_{\text{O\_CUSTKEY=C\_CUSTKEY}}$

Exercise 1

Determine the optimized execution tree with respect to the heuristic criteria presented in class

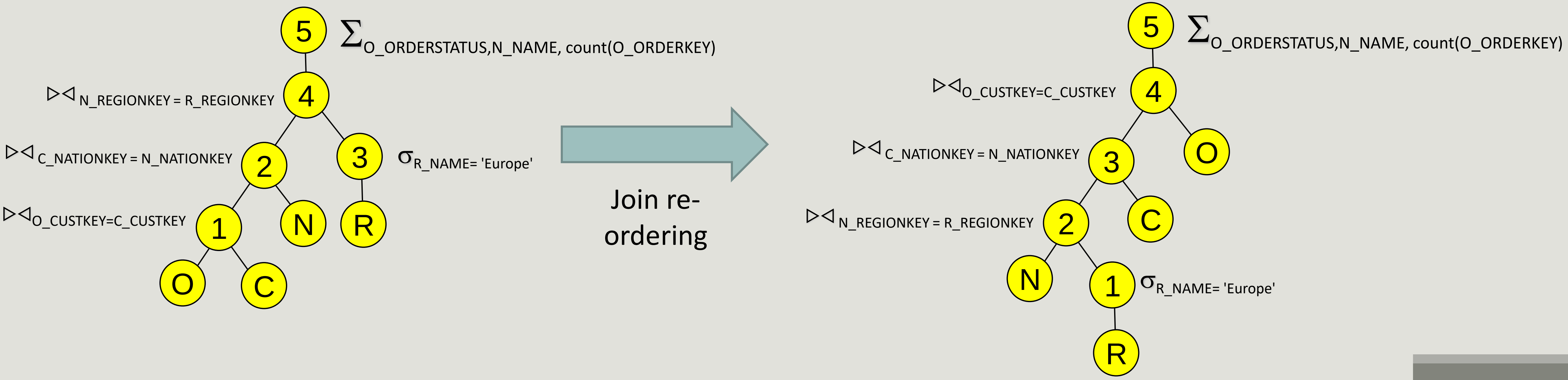

8 $\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$
7 $\sigma_{\text{N\_REGIONKEY = R\_REGIONKEY}}$
× 6
4 $\sigma_{\text{C\_NATIONKEY = N\_NATIONKEY}}$
5 $\sigma_{\text{R\_NAME= 'Europe'}}$
R
× 3
N
2 $\sigma_{\text{O\_CUSTKEY=C\_CUSTKEY}}$
× 1
O
C
5 $\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$
4 $\bowtie_{\text{N\_REGIONKEY = R\_REGIONKEY}}$
2 $\bowtie_{\text{C\_NATIONKEY = N\_NATIONKEY}}$
3 $\sigma_{\text{R\_NAME= 'Europe'}}$
1 $\bowtie_{\text{O\_CUSTKEY=C\_CUSTKEY}}$
N
R
O
C

Cartesian prod. + filtering replacement

Exercise 1

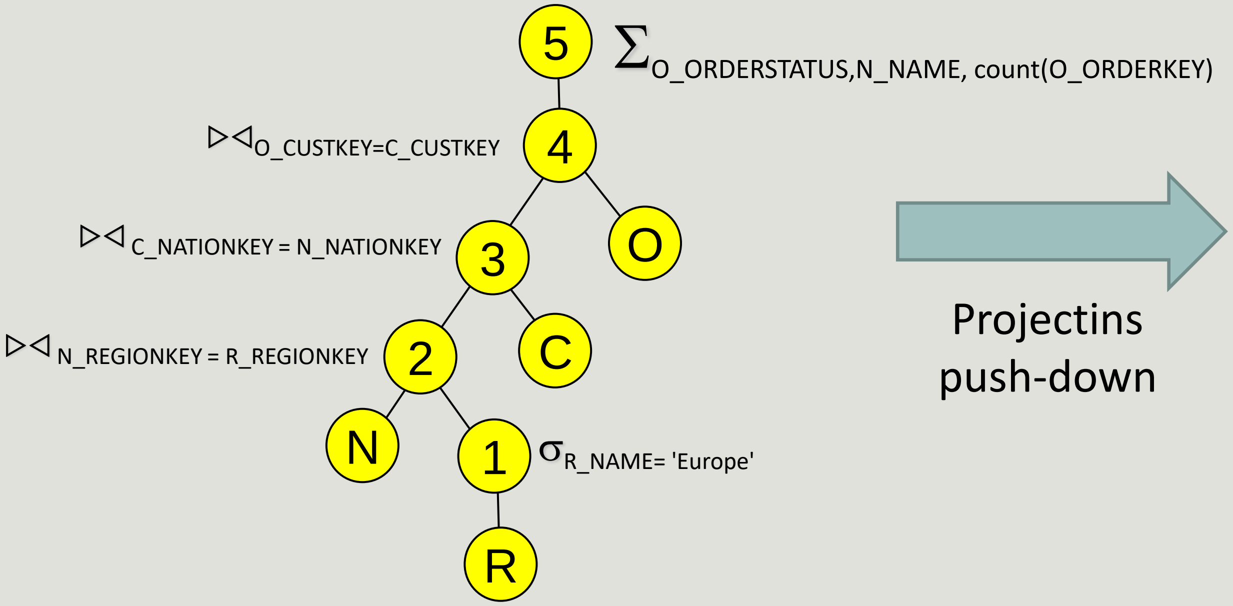
Determine the optimized execution tree with respect to the heuristic criteria presented in class

- Not all the orderings are feasible: it is necessary to build a join sequence. For example:
  R-N-O-C needs the cartesian product in R-**N-O**


5
Σ O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
▷◁ N_REGIONKEY = R_REGIONKEY
4
▷◁ C_NATIONKEY = N_NATIONKEY
2
3
σ R_NAME= 'Europe'
▷◁ O_CUSTKEY=C_CUSTKEY
1
N
R
O
C
Join re-ordering
5
Σ O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
▷◁ O_CUSTKEY=C_CUSTKEY
4
O
▷◁ C_NATIONKEY = N_NATIONKEY
3
C
▷◁ N_REGIONKEY = R_REGIONKEY
2
N
1
σ R_NAME= 'Europe'
R

Exercise 1

Determine the optimized execution tree with respect to the heuristic criteria presented in class



5 $\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$

4 $\bowtie_{\text{O\_CUSTKEY=C\_CUSTKEY}}$

3 $\bowtie_{\text{C\_NATIONKEY = N\_NATIONKEY}}$

O

2 $\bowtie_{\text{N\_REGIONKEY = R\_REGIONKEY}}$

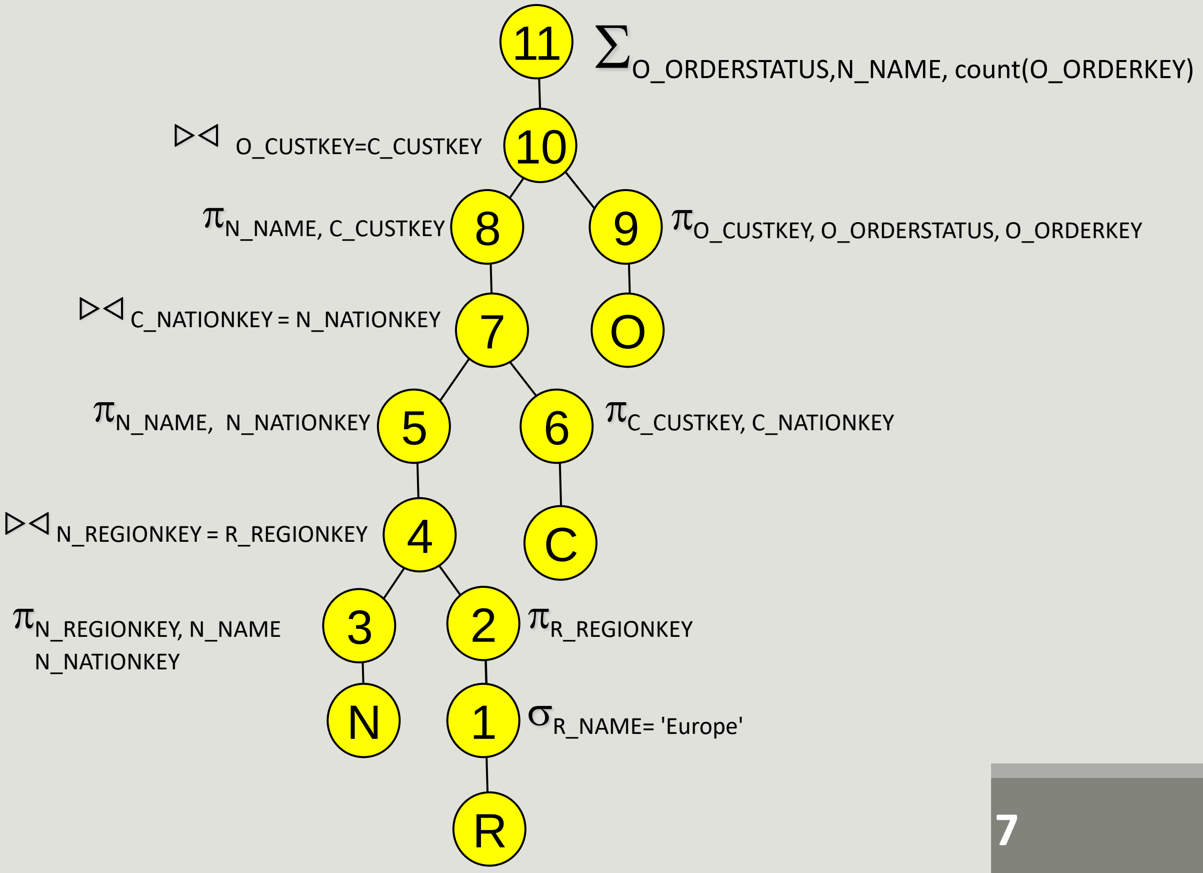
C

N

1 $\sigma_{\text{R\_NAME= 'Europe'}}$

R

Projectins push-down



11 $\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$

10 $\bowtie$ O_CUSTKEY=C_CUSTKEY

8 $\pi_{\text{N\_NAME, C\_CUSTKEY}}$

9 $\pi_{\text{O\_CUSTKEY, O\_ORDERSTATUS, O\_ORDERKEY}}$

7 $\bowtie_{\text{C\_NATIONKEY = N\_NATIONKEY}}$

O

5 $\pi_{\text{N\_NAME, N\_NATIONKEY}}$

6 $\pi_{\text{C\_CUSTKEY, C\_NATIONKEY}}$

4 $\bowtie_{\text{N\_REGIONKEY = R\_REGIONKEY}}$

C

3 $\pi_{\text{N\_REGIONKEY, N\_NAME}}$ N_NATIONKEY

2 $\pi_{\text{R\_REGIONKEY}}$

N

1 $\sigma_{\text{R\_NAME= 'Europe'}}$

R

Exercise 2

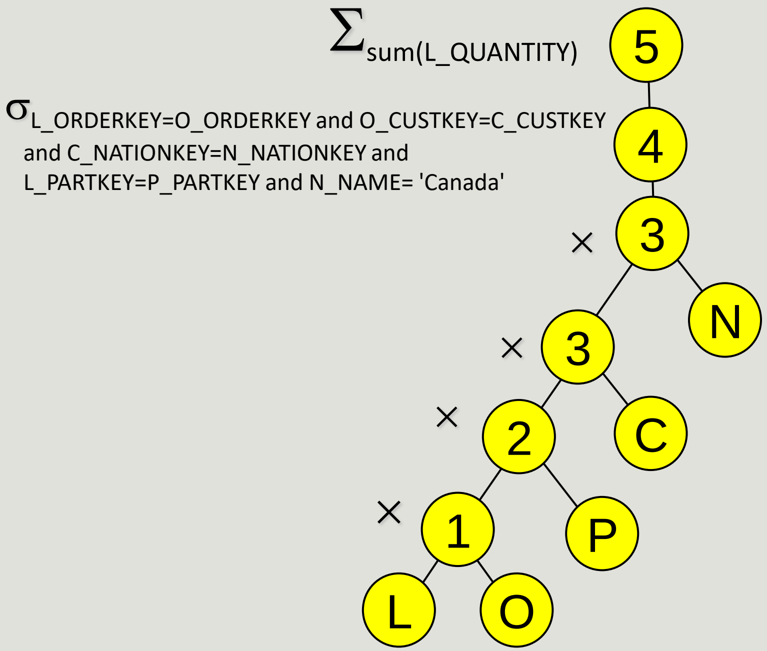
Determine the optimized execution tree with respect to the heuristic criteria presented in class

```
select  sum(L_QUANTITY)
from    LINEITEM,ORDERS,PART,CUSTOMER,NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY
    and C_NATIONKEY=N_NATIONKEY
    and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

Exercise 2

Determine the optimized execution tree with respect to the heuristic criteria presented in class

```
select  sum(L_QUANTITY)
from    LINEITEM,ORDERS,PART,CUSTOMER,NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY
    and C_NATIONKEY=N_NATIONKEY
    and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

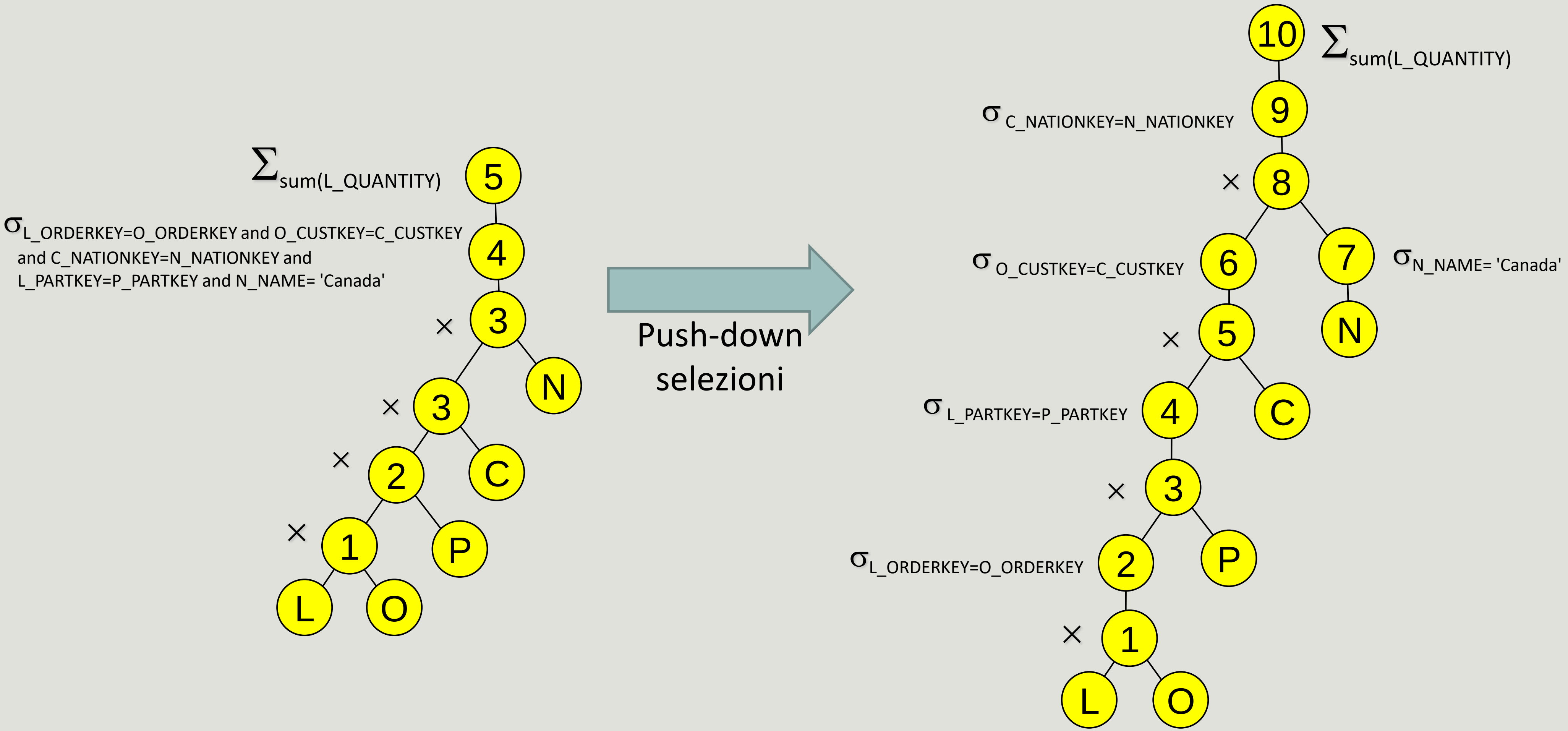


$\Sigma_{\text{sum(L\_QUANTITY)}}$

$\sigma_{\text{L\_ORDERKEY=O\_ORDERKEY and O\_CUSTKEY=C\_CUSTKEY}}$
and C_NATIONKEY=N_NATIONKEY and
L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'

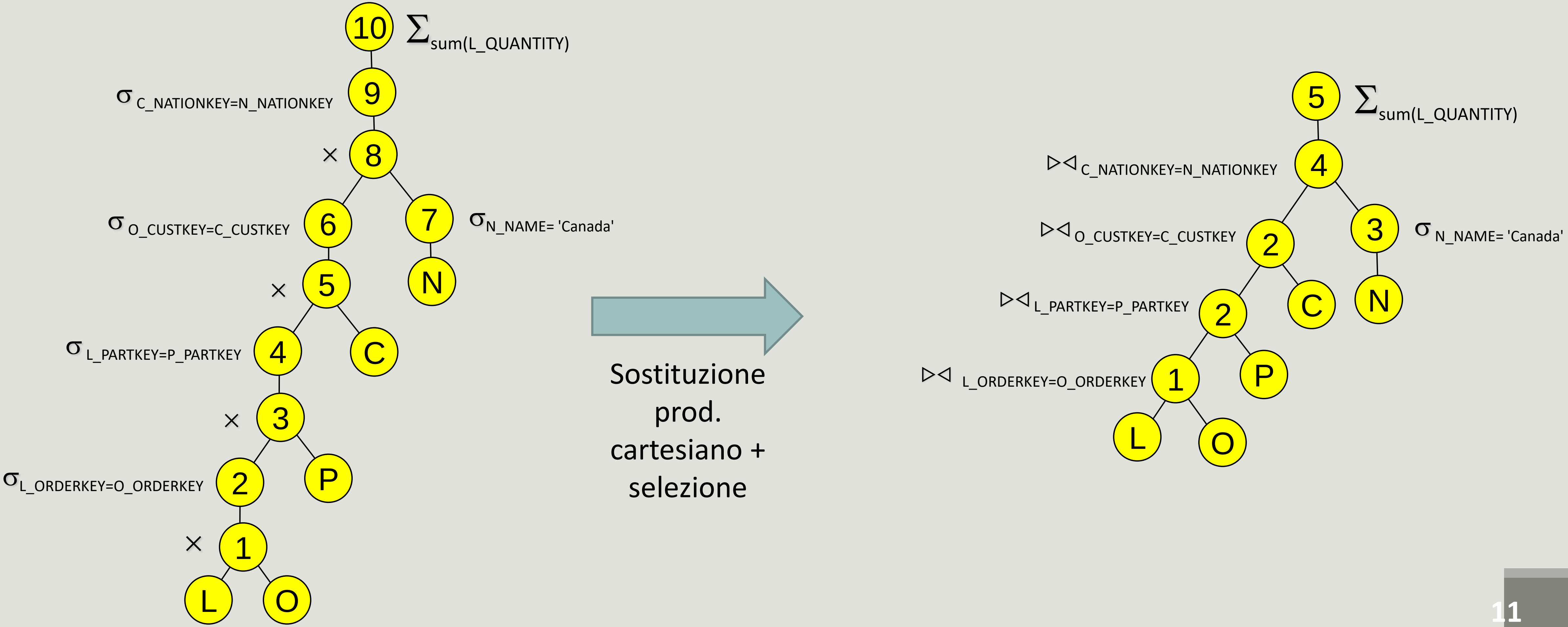
Exercise 2

Determine the optimized execution tree with respect to the heuristic criteria presented in class



Exercise 2

Determine the optimized execution tree with respect to the heuristic criteria presented in class

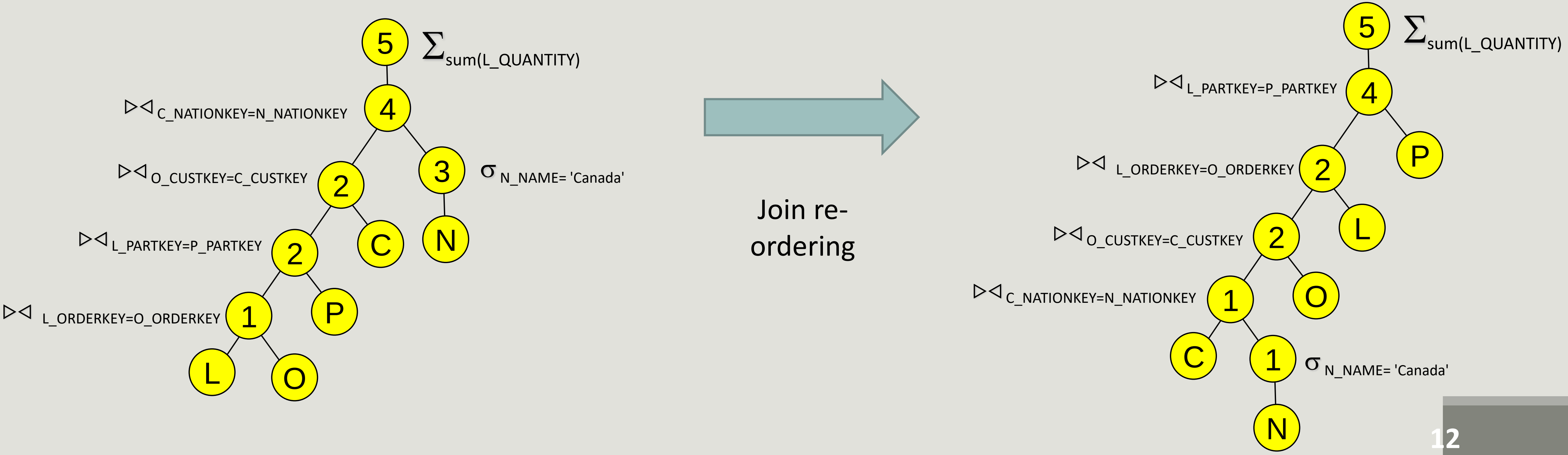

10 $\Sigma_{sum(L\_QUANTITY)}$
9 $\sigma_{C\_NATIONKEY=N\_NATIONKEY}$
8 ×
6 $\sigma_{O\_CUSTKEY=C\_CUSTKEY}$
7 $\sigma_{N\_NAME= 'Canada'}$
N
5 ×
C
4 $\sigma_{L\_PARTKEY=P\_PARTKEY}$
3 ×
P
2 $\sigma_{L\_ORDERKEY=O\_ORDERKEY}$
1 ×
L
O
5 $\Sigma_{sum(L\_QUANTITY)}$
4 $\bowtie_{C\_NATIONKEY=N\_NATIONKEY}$
2 $\bowtie_{O\_CUSTKEY=C\_CUSTKEY}$
3 $\sigma_{N\_NAME= 'Canada'}$
2 $\bowtie_{L\_PARTKEY=P\_PARTKEY}$
C
N
1 $\bowtie_{L\_ORDERKEY=O\_ORDERKEY}$
P
L
O

Sostituzione prod. cartesiano + selezione

Exercise 2

Determine the optimized execution tree with respect to the heuristic criteria presented in class

- Not all orderings are feasible: a join sequence is needed. For example C-N-O-P-L requires a cartesian product for C-N-O-**P**
- Projection push-down is omitted for brevity


5 $\Sigma_{sum(L\_QUANTITY)}$
4 ▷◁ C_NATIONKEY=N_NATIONKEY
2 ▷◁ O_CUSTKEY=C_CUSTKEY
3 σ N_NAME= 'Canada'
2 ▷◁ L_PARTKEY=P_PARTKEY
C
N
1 ▷◁ L_ORDERKEY=O_ORDERKEY
P
L
O
Join re-ordering
5 $\Sigma_{sum(L\_QUANTITY)}$
4 ▷◁ L_PARTKEY=P_PARTKEY
P
2 ▷◁ L_ORDERKEY=O_ORDERKEY
L
2 ▷◁ O_CUSTKEY=C_CUSTKEY
O
1 ▷◁ C_NATIONKEY=N_NATIONKEY
C
1 σ N_NAME= 'Canada'
N

Exercise 3

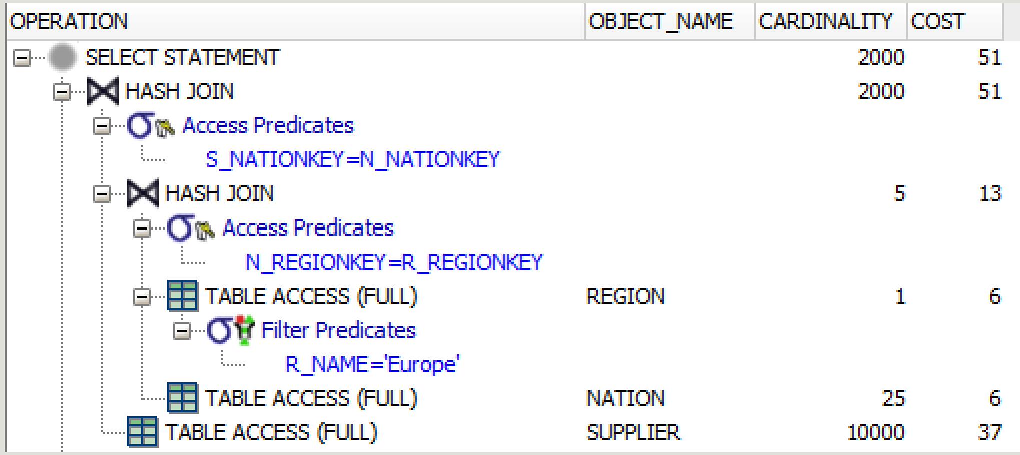
Draw the execution tree proposed by ORACLE for the following queries:

```
select  s_name, s_address
from    TPCD.supplier, TPCD.nation, TPCD.region
where   s_nationkey = n_nationkey and n_regionkey = r_regionkey and r_name=’Europe’;
```

Exercise 3

Draw the execution tree proposed by ORACLE for the following queries:

```
select  s_name, s_address
from    TPCD.supplier, TPCD.nation, TPCD.region
where   s_nationkey = n_nationkey and n_regionkey = r_regionkey and r_name='Europe';
```



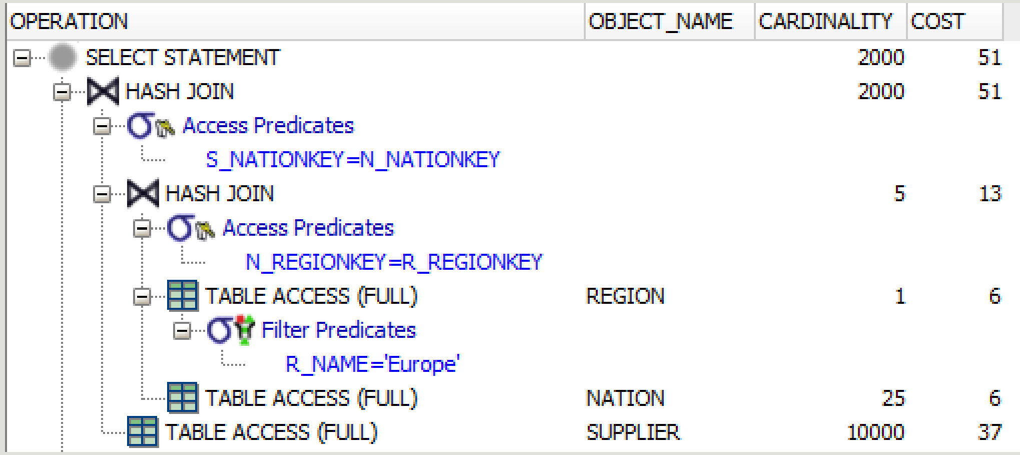
```
OPERATION                          OBJECT_NAME  CARDINALITY  COST
SELECT STATEMENT                                       2000    51
  HASH JOIN                                            2000    51
    Access Predicates
      S_NATIONKEY=N_NATIONKEY
    HASH JOIN                                             5    13
      Access Predicates
        N_REGIONKEY=R_REGIONKEY
      TABLE ACCESS (FULL)          REGION                 1     6
        Filter Predicates
          R_NAME='Europe'
      TABLE ACCESS (FULL)          NATION                25     6
    TABLE ACCESS (FULL)            SUPPLIER           10000    37
```

The DBMS omits the projections, we do not report them too

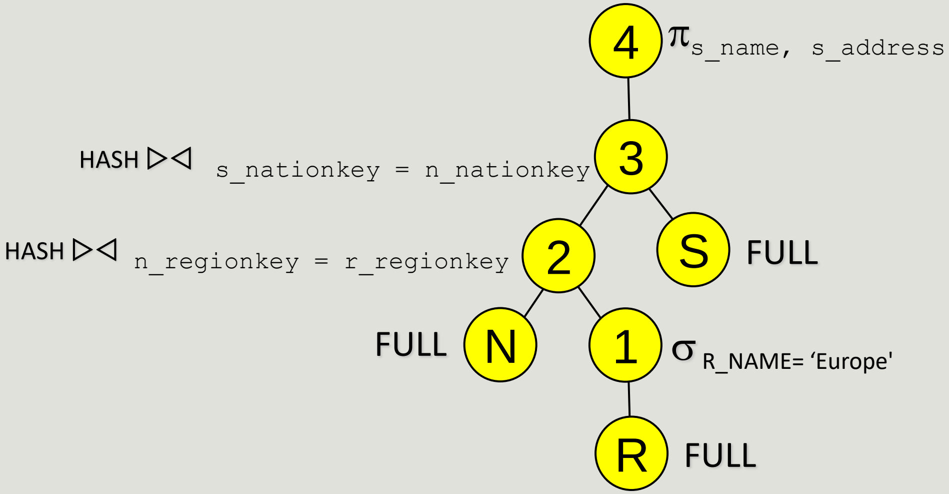
Exercise 3

Draw the execution tree proposed by ORACLE for the following queries:

```
select  s_name, s_address
from    TPCD.supplier, TPCD.nation, TPCD.region
where   s_nationkey = n_nationkey and n_regionkey = r_regionkey and r_name='Europe';
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		2000	51
HASH JOIN		2000	51
Access Predicates			
S_NATIONKEY=N_NATIONKEY			
HASH JOIN		5	13
Access Predicates			
N_REGIONKEY=R_REGIONKEY			
TABLE ACCESS (FULL)	REGION	1	6
Filter Predicates			
R_NAME='Europe'			
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	SUPPLIER	10000	37


4
$\pi_{s\_name,\ s\_address}$
HASH ▷◁ s_nationkey = n_nationkey
3
HASH ▷◁ n_regionkey = r_regionkey
2
S
FULL
FULL
N
1
$\sigma_{R\_NAME=\text{'Europe'}}$
R
FULL

The DBMS omits the projections, we do not report them too

Exercise 4 (see 1)

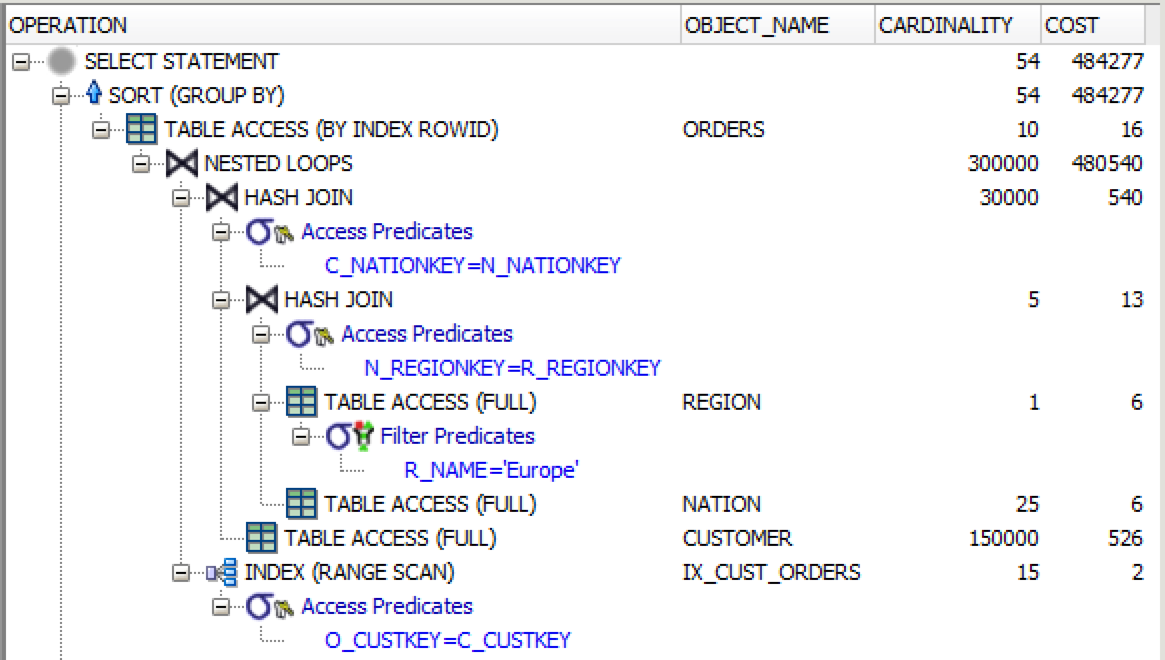
Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```

Exercise 4 (see 1)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```

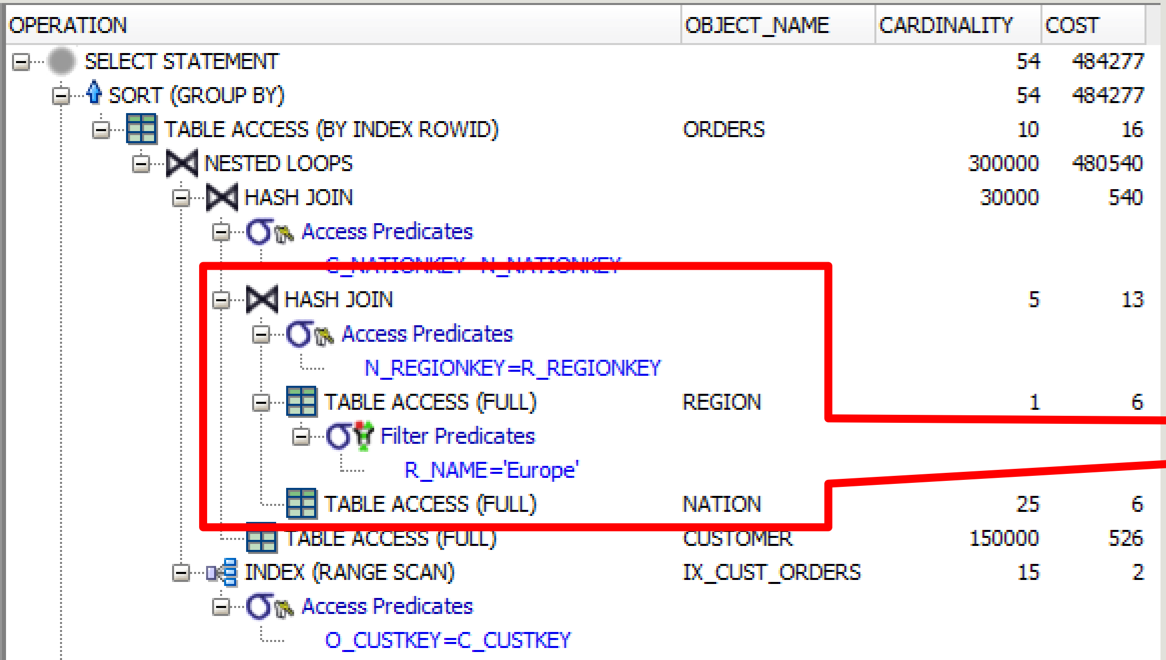


```
OPERATION                                   OBJECT_NAME     CARDINALITY   COST
SELECT STATEMENT                                                     54  484277
  SORT (GROUP BY)                                                    54  484277
    TABLE ACCESS (BY INDEX ROWID)           ORDERS                   10      16
      NESTED LOOPS                                               300000  480540
        HASH JOIN                                                 30000     540
          Access Predicates
            C_NATIONKEY=N_NATIONKEY
          HASH JOIN                                                   5      13
            Access Predicates
              N_REGIONKEY=R_REGIONKEY
            TABLE ACCESS (FULL)             REGION                    1       6
              Filter Predicates
                R_NAME='Europe'
            TABLE ACCESS (FULL)             NATION                   25       6
          TABLE ACCESS (FULL)               CUSTOMER             150000     526
        INDEX (RANGE SCAN)                  IX_CUST_ORDERS           15       2
          Access Predicates
            O_CUSTKEY=C_CUSTKEY
```

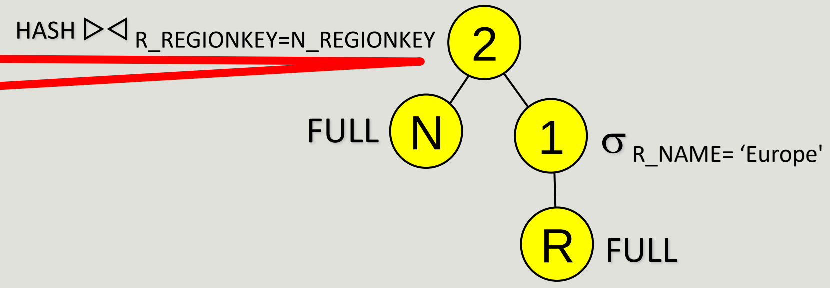
Exercise 4 (see 1)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```



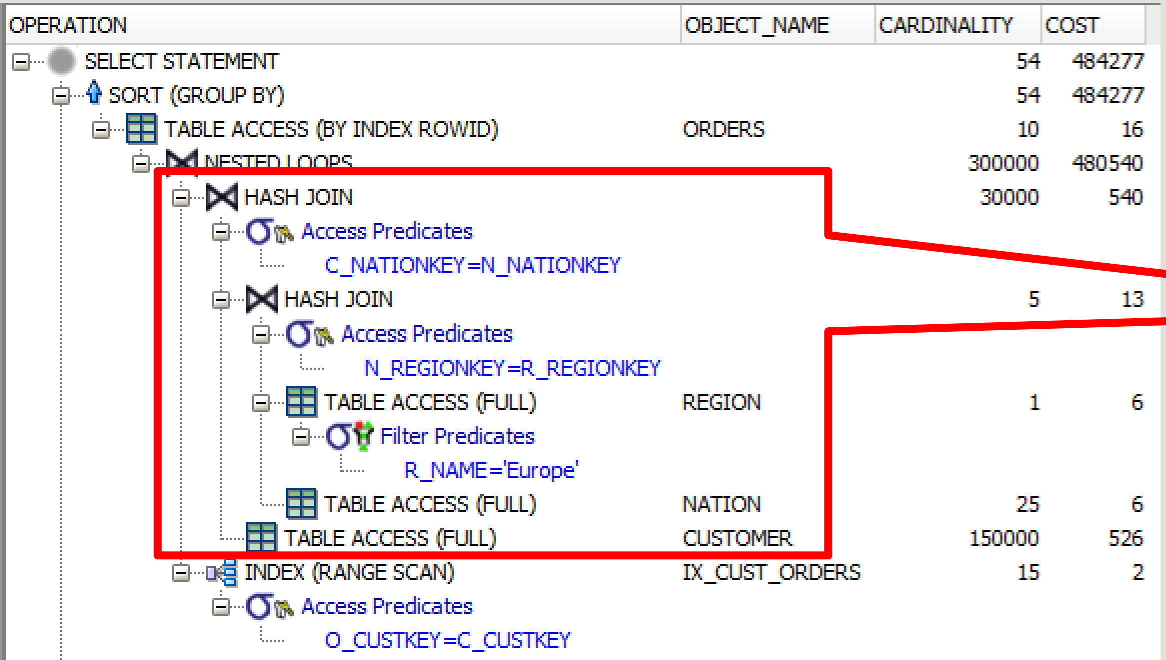
```
OPERATION                                         OBJECT_NAME    CARDINALITY   COST
SELECT STATEMENT                                                          54  484277
  SORT (GROUP BY)                                                         54  484277
    TABLE ACCESS (BY INDEX ROWID)                 ORDERS                  10      16
      NESTED LOOPS                                                    300000  480540
        HASH JOIN                                                      30000     540
          Access Predicates
            C_NATIONKEY=N_NATIONKEY
          HASH JOIN                                                        5      13
            Access Predicates
              N_REGIONKEY=R_REGIONKEY
            TABLE ACCESS (FULL)                   REGION                   1       6
              Filter Predicates
                R_NAME='Europe'
            TABLE ACCESS (FULL)                   NATION                  25       6
          TABLE ACCESS (FULL)                     CUSTOMER            150000     526
        INDEX (RANGE SCAN)                        IX_CUST_ORDERS          15       2
          Access Predicates
            O_CUSTKEY=C_CUSTKEY
```



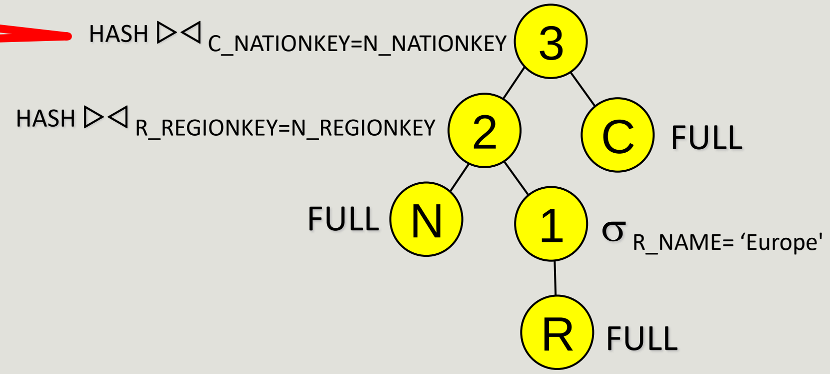
Exercise 4 (see 1)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```



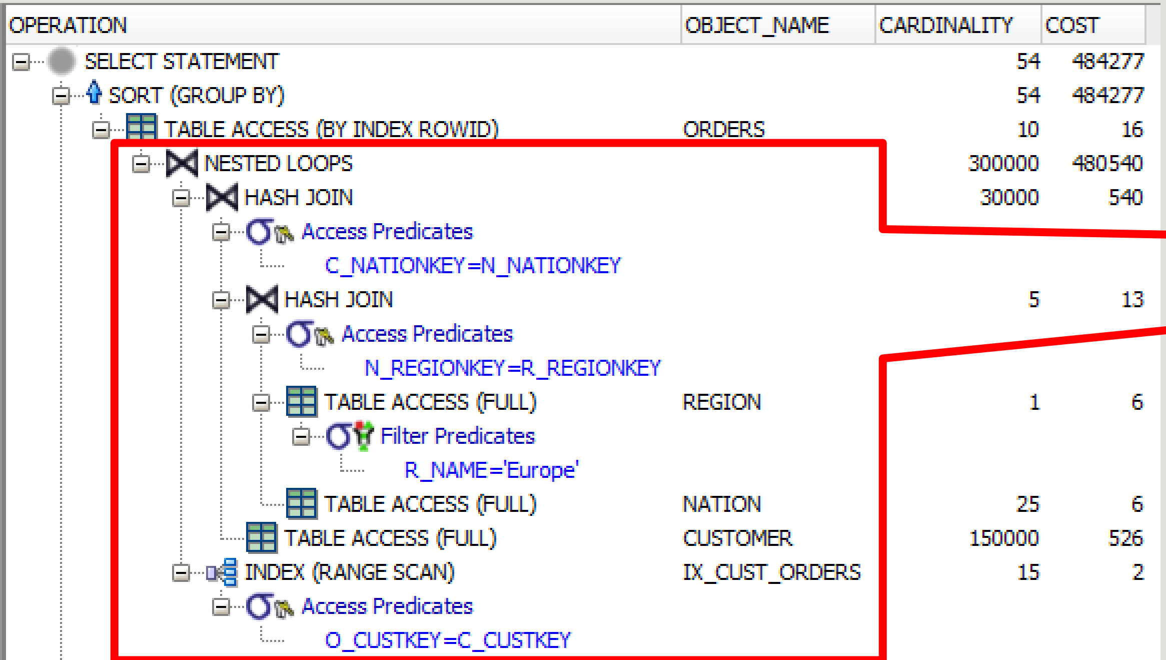
```
OPERATION                                  OBJECT_NAME     CARDINALITY   COST
SELECT STATEMENT                                                    54  484277
  SORT (GROUP BY)                                                   54  484277
    TABLE ACCESS (BY INDEX ROWID)          ORDERS                   10      16
      NESTED LOOPS                                              300000  480540
        HASH JOIN                                                30000     540
          Access Predicates
            C_NATIONKEY=N_NATIONKEY
          HASH JOIN                                                  5      13
            Access Predicates
              N_REGIONKEY=R_REGIONKEY
            TABLE ACCESS (FULL)            REGION                    1       6
              Filter Predicates
                R_NAME='Europe'
            TABLE ACCESS (FULL)            NATION                   25       6
          TABLE ACCESS (FULL)              CUSTOMER             150000     526
        INDEX (RANGE SCAN)                 IX_CUST_ORDERS           15       2
          Access Predicates
            O_CUSTKEY=C_CUSTKEY
```


HASH ▷◁ C_NATIONKEY=N_NATIONKEY
3
HASH ▷◁ R_REGIONKEY=N_REGIONKEY
2
C
FULL
FULL
N
1
σ R_NAME= 'Europe'
R
FULL

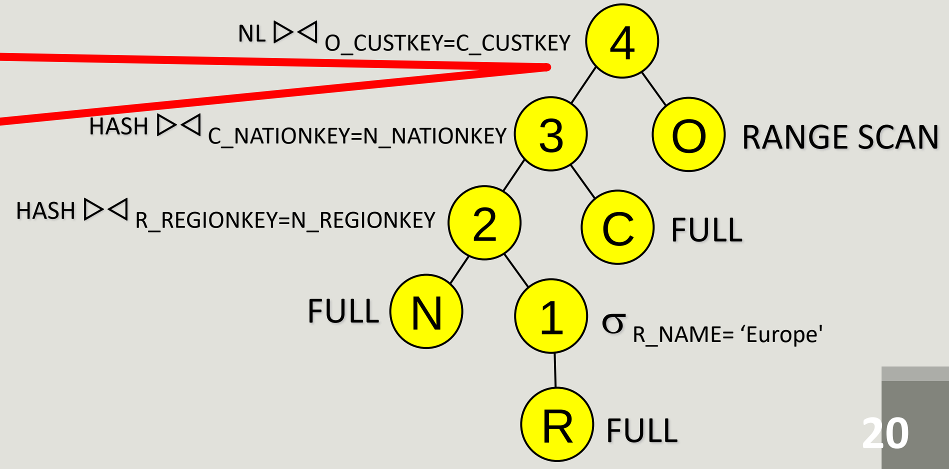
Exercise 4 (see 1)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```



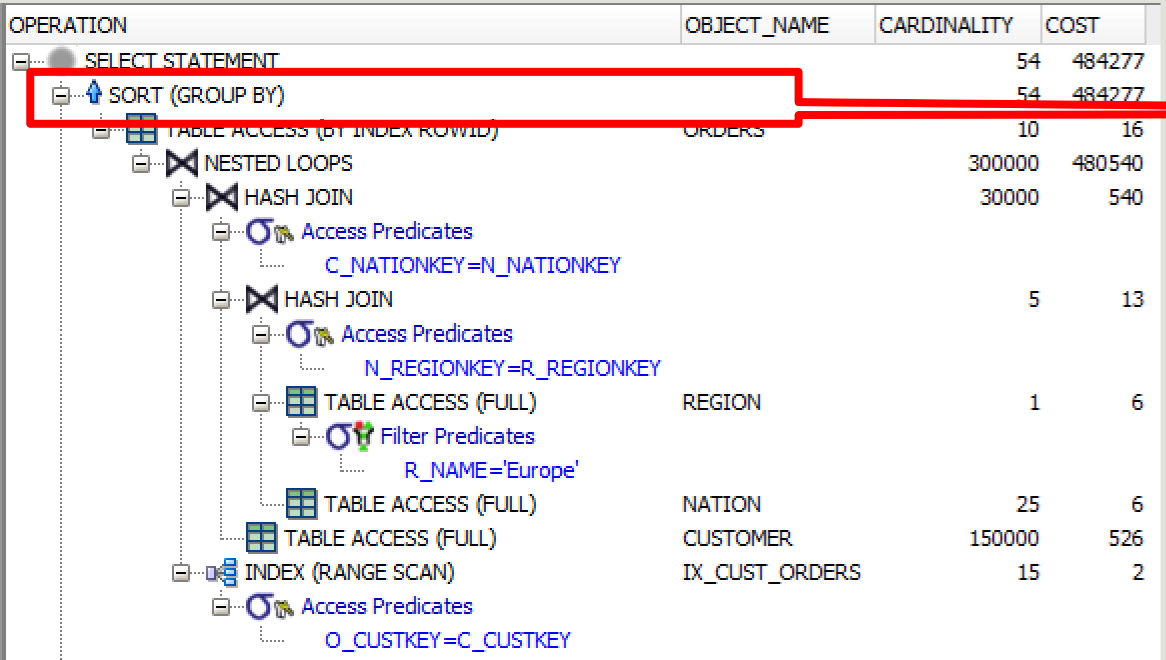
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		54	484277
SORT (GROUP BY)		54	484277
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
NESTED LOOPS		300000	480540
HASH JOIN		30000	540
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
HASH JOIN		5	13
Access Predicates			
N_REGIONKEY=R_REGIONKEY			
TABLE ACCESS (FULL)	REGION	1	6
Filter Predicates			
R_NAME='Europe'			
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	CUSTOMER	150000	526
INDEX (RANGE SCAN)	IX_CUST_ORDERS	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			


NL ▷◁ O_CUSTKEY=C_CUSTKEY
4
HASH ▷◁ C_NATIONKEY=N_NATIONKEY
3
O
RANGE SCAN
HASH ▷◁ R_REGIONKEY=N_REGIONKEY
2
C
FULL
FULL
N
1
σ R_NAME= 'Europe'
R
FULL

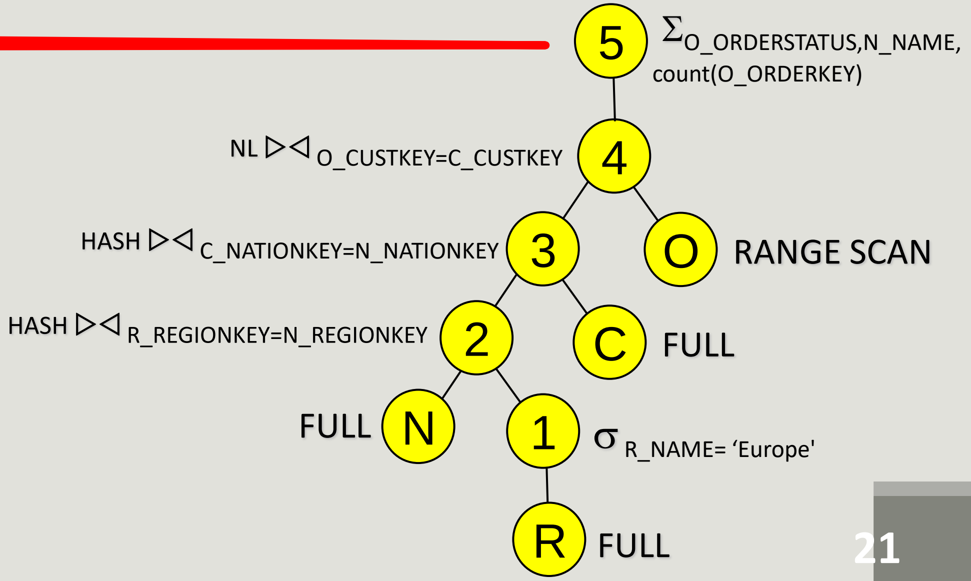
Exercise 4 (see 1)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		54	484277
SORT (GROUP BY)		54	484277
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
NESTED LOOPS		300000	480540
HASH JOIN		30000	540
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
HASH JOIN		5	13
Access Predicates			
N_REGIONKEY=R_REGIONKEY			
TABLE ACCESS (FULL)	REGION	1	6
Filter Predicates			
R_NAME='Europe'			
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	CUSTOMER	150000	526
INDEX (RANGE SCAN)	IX_CUST_ORDERS	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			

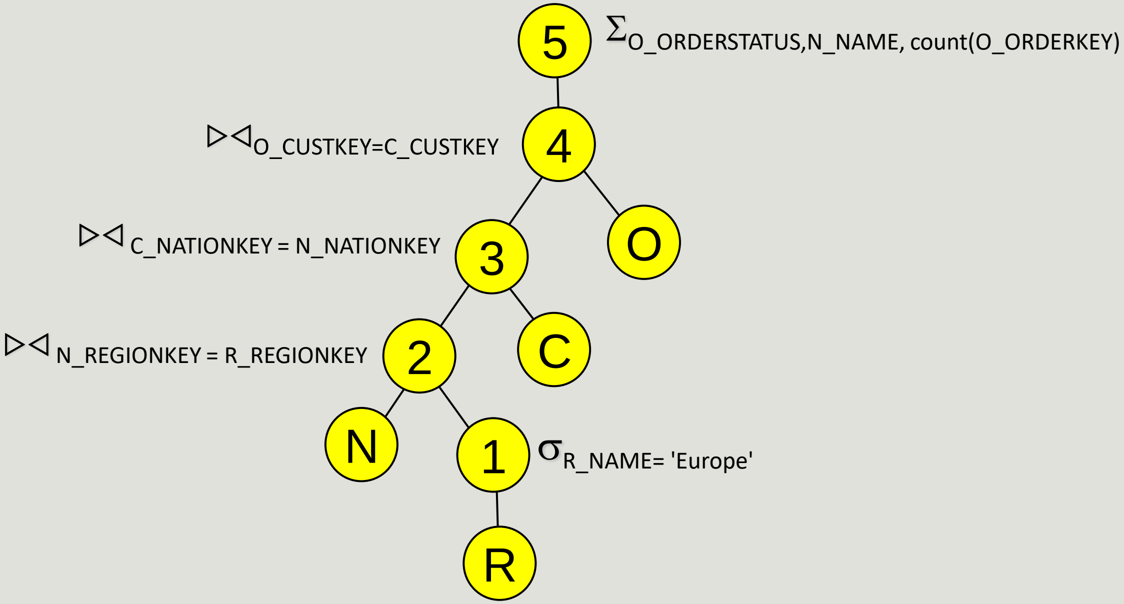


5
$\Sigma_{\text{O\_ORDERSTATUS,N\_NAME, count(O\_ORDERKEY)}}$
NL ▷◁ O_CUSTKEY=C_CUSTKEY
4
HASH ▷◁ C_NATIONKEY=N_NATIONKEY
3
O
RANGE SCAN
HASH ▷◁ R_REGIONKEY=N_REGIONKEY
2
C
FULL
FULL
N
1
σ R_NAME= 'Europe'
R
FULL

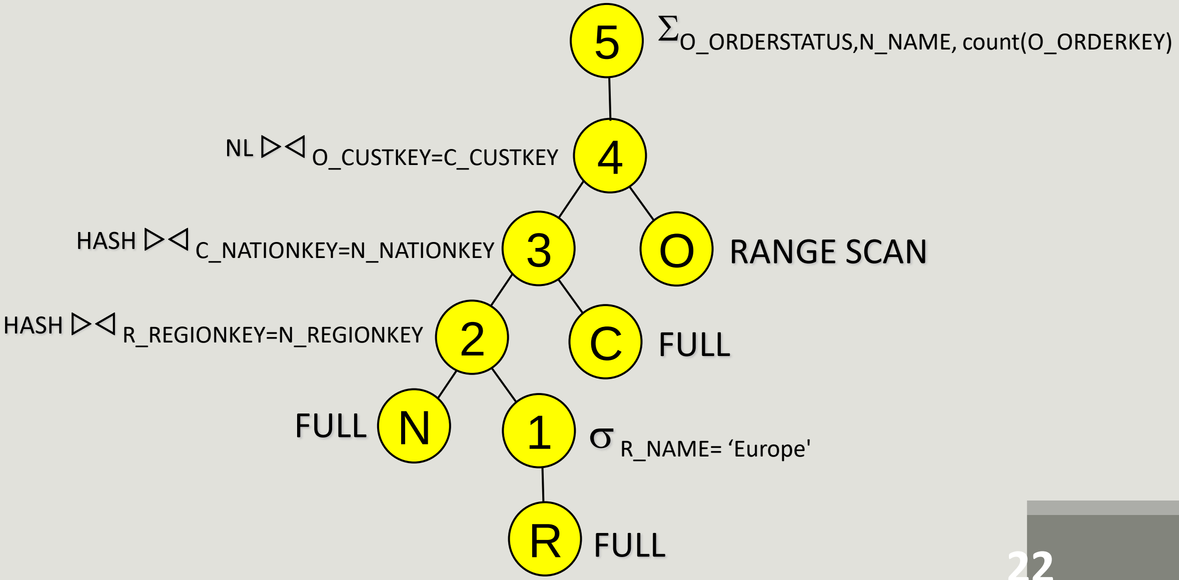
Heuristic plan (see 1) vs Oracle plan

Draw the execution tree proposed by ORACLE for the following queries:

```
select  O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
from    TPCD.ORDERS, TPCD.CUSTOMER, TPCD.NATION, TPCD.REGION
where   O_CUSTKEY=C_CUSTKEY and C_NATIONKEY = N_NATIONKEY
    and N_REGIONKEY = R_REGIONKEY and R_NAME= 'Europe'
group by O_ORDERSTATUS,N_NAME
```



5
Σ O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
▷◁ O_CUSTKEY=C_CUSTKEY
4
▷◁ C_NATIONKEY = N_NATIONKEY
3
O
▷◁ N_REGIONKEY = R_REGIONKEY
2
C
N
1
σ R_NAME= 'Europe'
R



5
Σ O_ORDERSTATUS,N_NAME, count(O_ORDERKEY)
NL ▷◁ O_CUSTKEY=C_CUSTKEY
4
HASH ▷◁ C_NATIONKEY=N_NATIONKEY
3
O
RANGE SCAN
HASH ▷◁ R_REGIONKEY=N_REGIONKEY
2
C
FULL
FULL
N
1
σ R_NAME= 'Europe'
R
FULL

Exercise 5 (see 2)

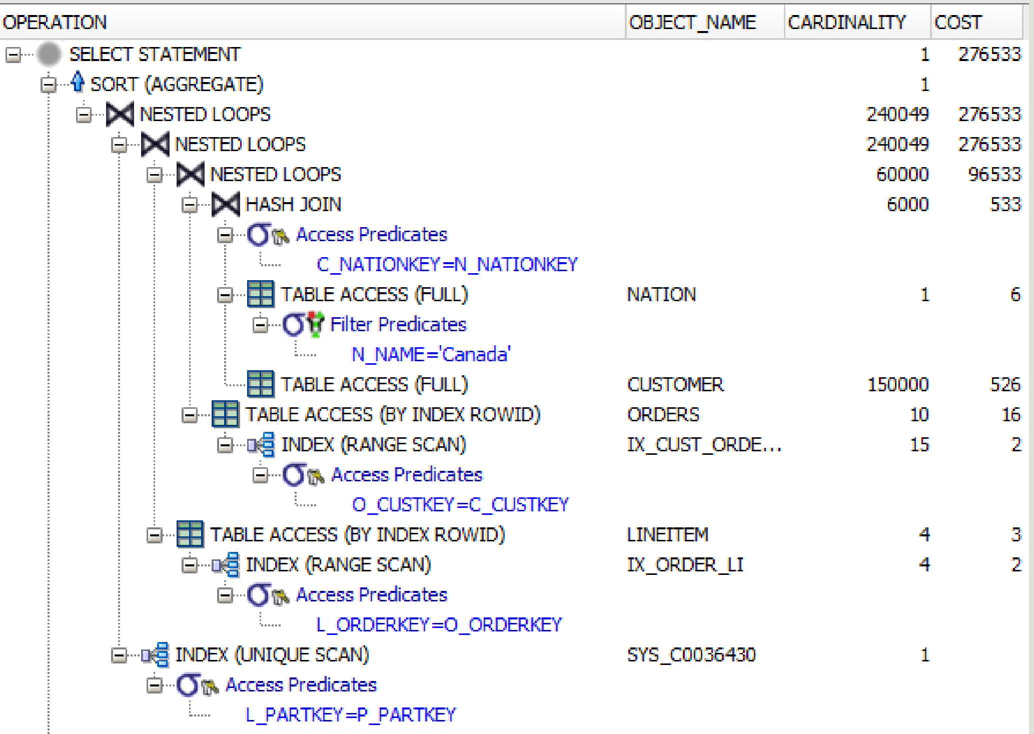
Draw the execution tree proposed by ORACLE for the following queries:

```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```



```
OPERATION                                  OBJECT_NAME     CARDINALITY    COST
SELECT STATEMENT                                                     1  276533
  SORT (AGGREGATE)                                                   1
    NESTED LOOPS                                                240049  276533
      NESTED LOOPS                                              240049  276533
        NESTED LOOPS                                             60000   96533
          HASH JOIN                                               6000     533
            Access Predicates
              C_NATIONKEY=N_NATIONKEY
            TABLE ACCESS (FULL)            NATION                    1       6
              Filter Predicates
                N_NAME='Canada'
            TABLE ACCESS (FULL)            CUSTOMER             150000     526
          TABLE ACCESS (BY INDEX ROWID)    ORDERS                   10      16
            INDEX (RANGE SCAN)             IX_CUST_ORDE...          15       2
              Access Predicates
                O_CUSTKEY=C_CUSTKEY
        TABLE ACCESS (BY INDEX ROWID)      LINEITEM                  4       3
          INDEX (RANGE SCAN)               IX_ORDER_LI               4       2
            Access Predicates
              L_ORDERKEY=O_ORDERKEY
      INDEX (UNIQUE SCAN)                  SYS_C0036430              1
        Access Predicates
          L_PARTKEY=P_PARTKEY
```

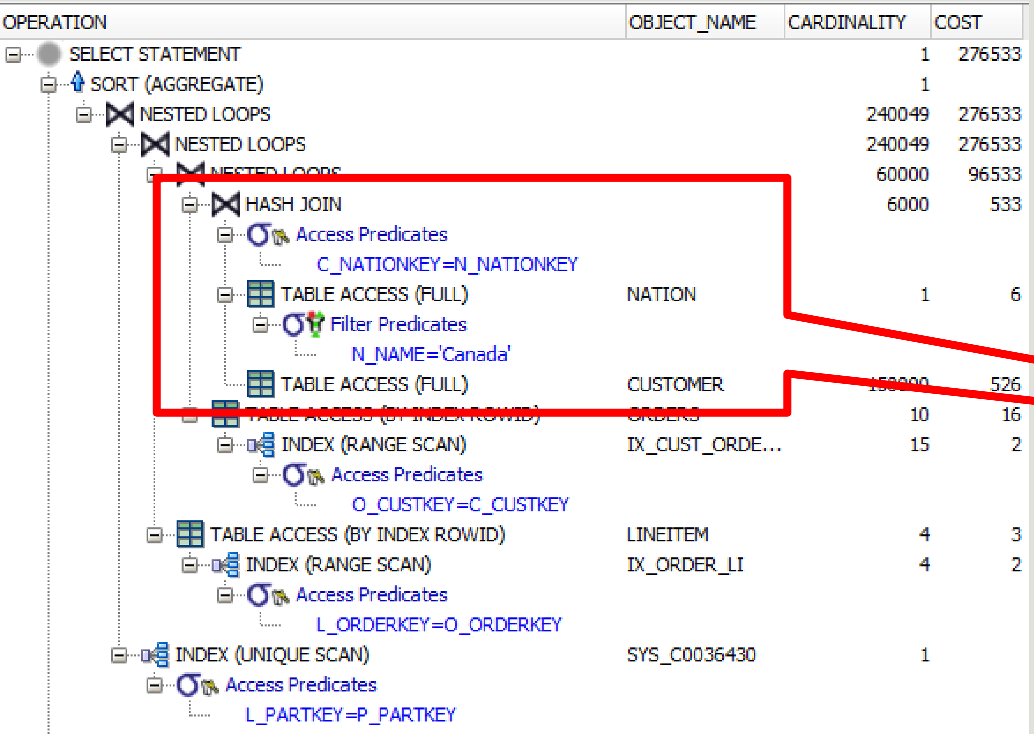
Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

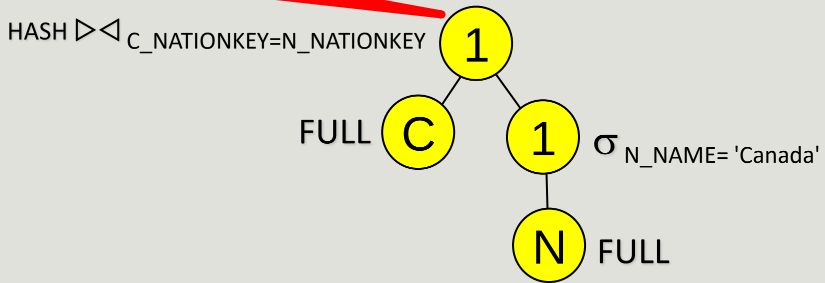
```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

ORACLE accesses the index but not the Part table. Why?

Actually there would be no need, why?



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	276533
SORT (AGGREGATE)		1	
NESTED LOOPS		240049	276533
NESTED LOOPS		240049	276533
NESTED LOOPS		60000	96533
HASH JOIN		6000	533
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='Canada'			
TABLE ACCESS (FULL)	CUSTOMER	150000	526
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
INDEX (RANGE SCAN)	IX_CUST_ORDE...	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			



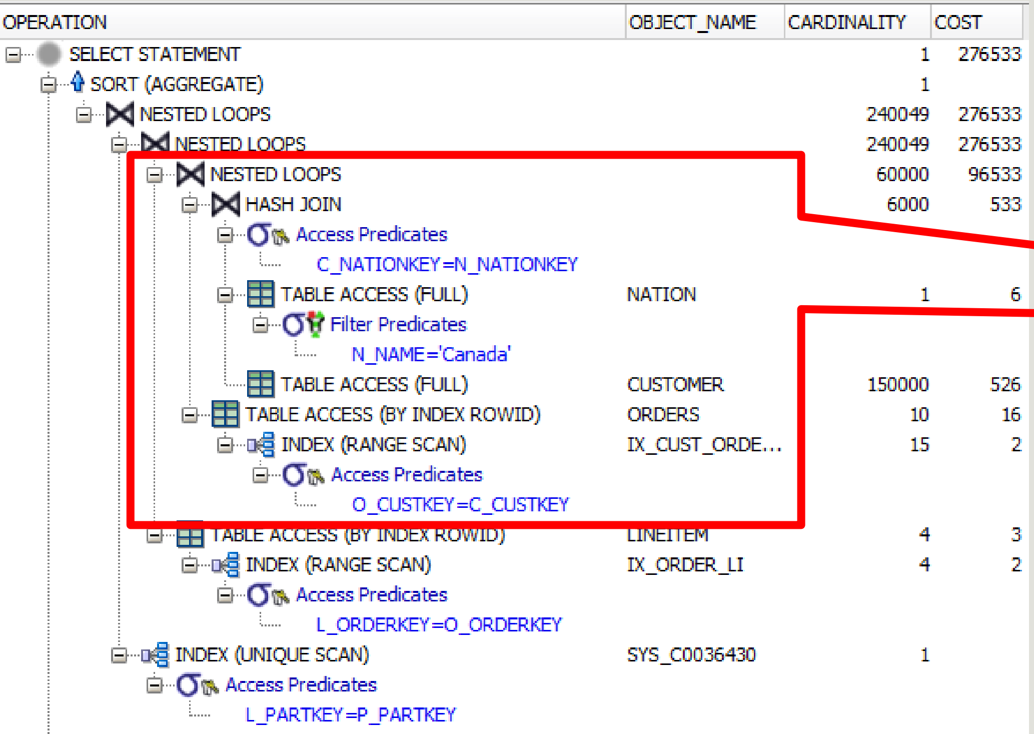
Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

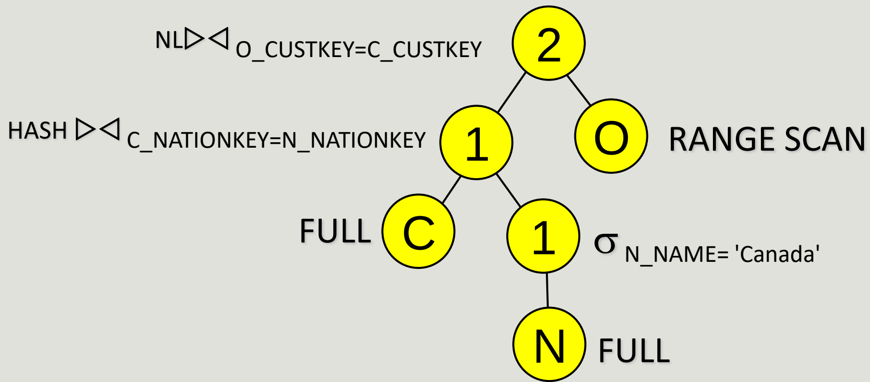
```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

ORACLE accesses the index but not the Part table. Why?

Actually there would be no need, why?



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	276533
SORT (AGGREGATE)		1	
NESTED LOOPS		240049	276533
NESTED LOOPS		240049	276533
NESTED LOOPS		60000	96533
HASH JOIN		6000	533
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='Canada'			
TABLE ACCESS (FULL)	CUSTOMER	150000	526
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
INDEX (RANGE SCAN)	IX_CUST_ORDE...	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			



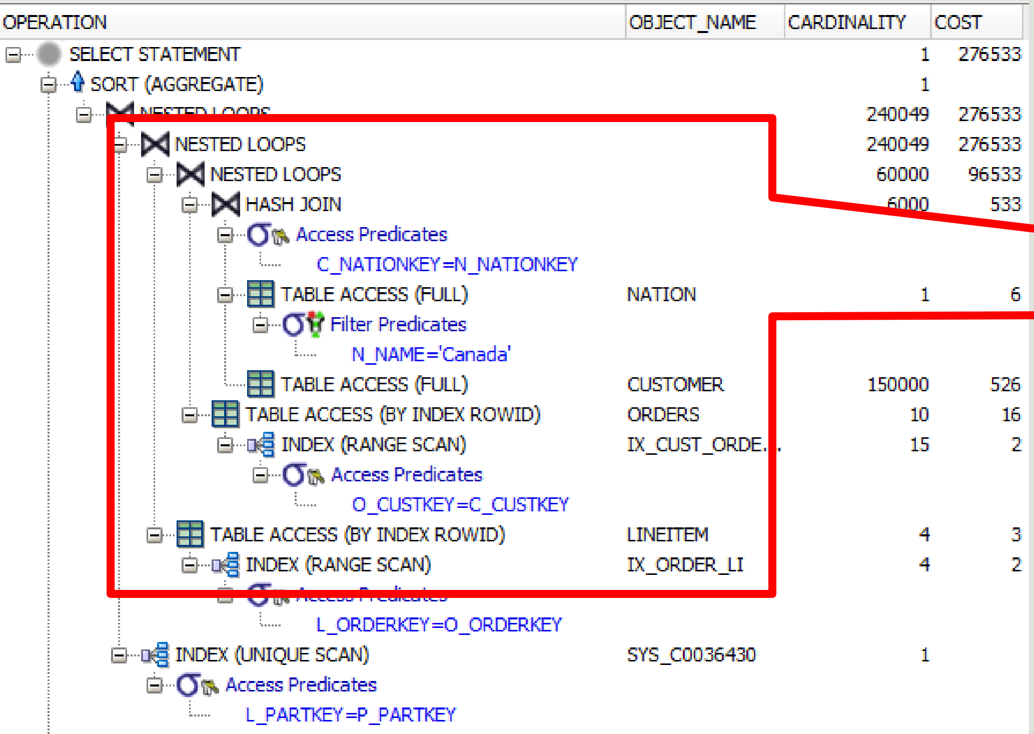
Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

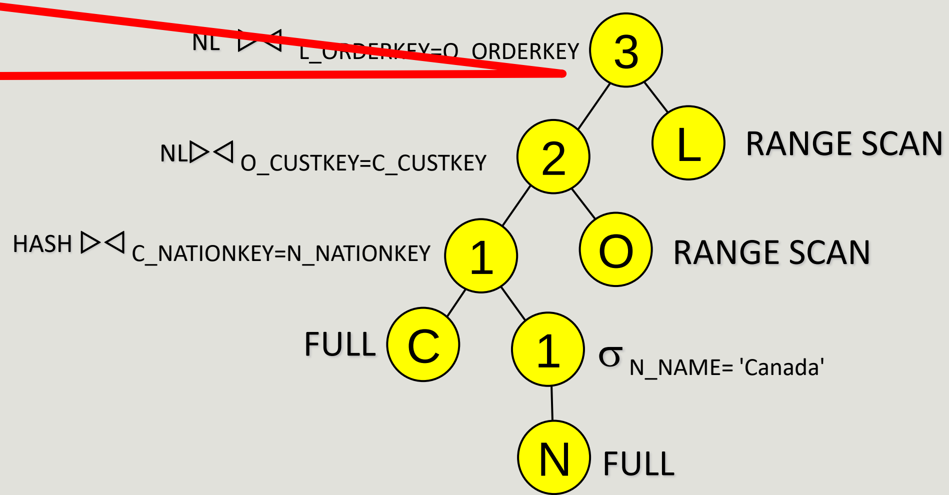
```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

ORACLE accesses the index but not the Part table. Why?

Actually there would be no need, why?



```
OPERATION                                      OBJECT_NAME     CARDINALITY   COST
SELECT STATEMENT                                                         1  276533
  SORT (AGGREGATE)                                                       1
    NESTED LOOPS                                                    240049  276533
      NESTED LOOPS                                                  240049  276533
        NESTED LOOPS                                                 60000   96533
          HASH JOIN                                                   6000     533
            Access Predicates
              C_NATIONKEY=N_NATIONKEY
            TABLE ACCESS (FULL)                NATION                    1       6
              Filter Predicates
                N_NAME='Canada'
            TABLE ACCESS (FULL)                CUSTOMER             150000     526
          TABLE ACCESS (BY INDEX ROWID)        ORDERS                   10      16
            INDEX (RANGE SCAN)                 IX_CUST_ORDE...          15       2
              Access Predicates
                O_CUSTKEY=C_CUSTKEY
        TABLE ACCESS (BY INDEX ROWID)          LINEITEM                  4       3
          INDEX (RANGE SCAN)                   IX_ORDER_LI               4       2
            Access Predicates
              L_ORDERKEY=O_ORDERKEY
      INDEX (UNIQUE SCAN)                      SYS_C0036430              1
        Access Predicates
          L_PARTKEY=P_PARTKEY
```



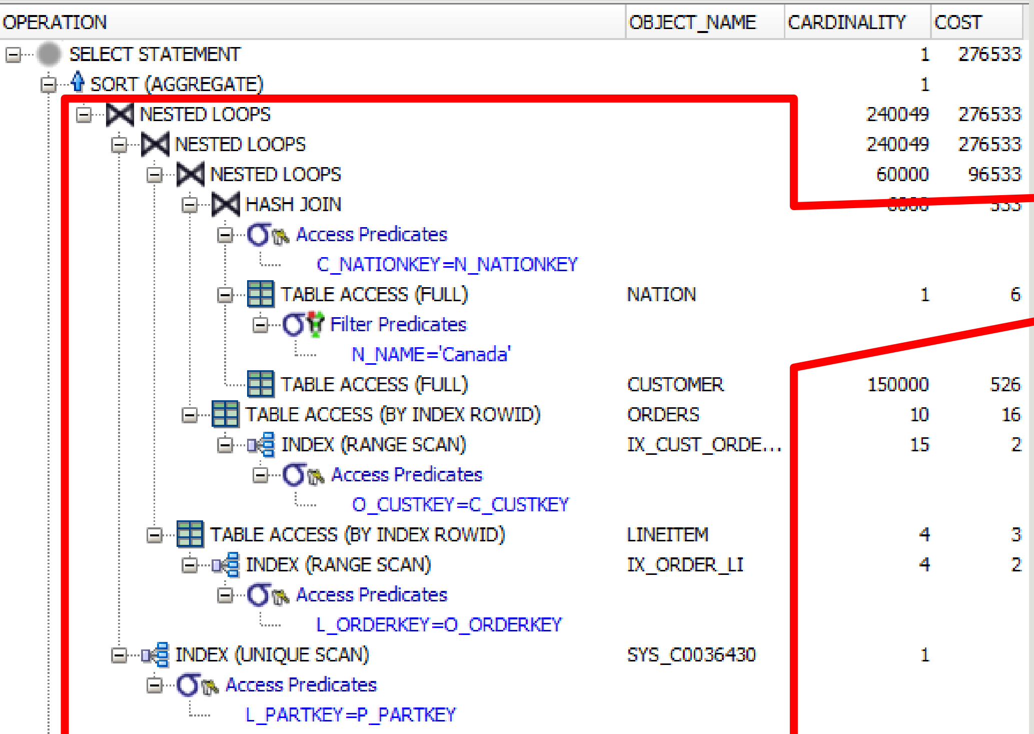
Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

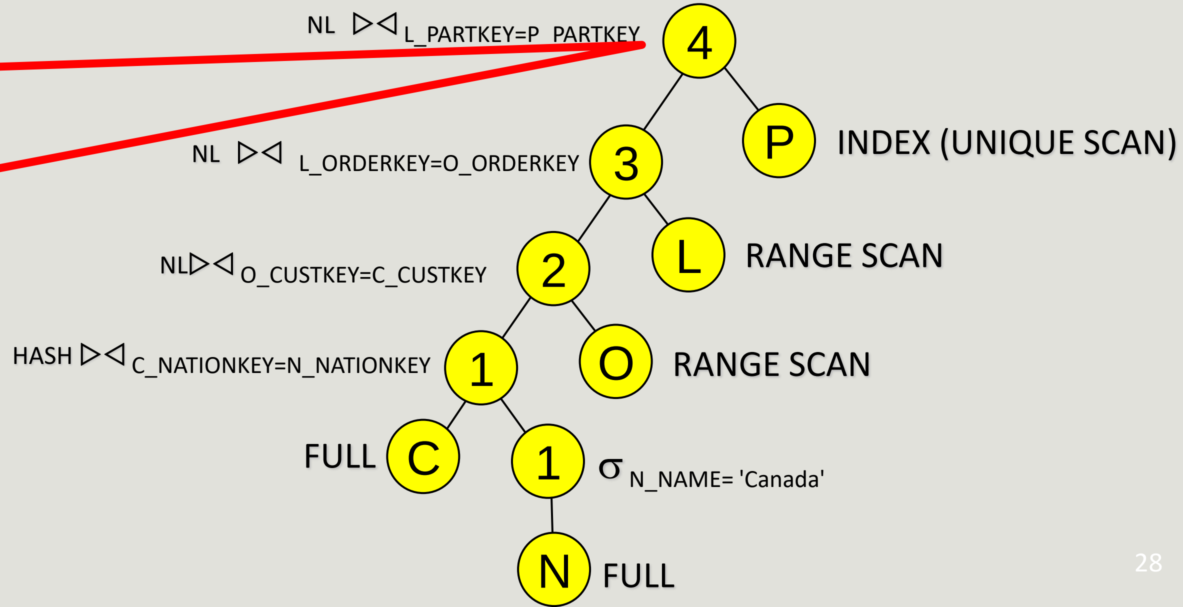
ORACLE accesses the index but not the Part table. Why?

Actually there would be no need, why?

```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	276533
SORT (AGGREGATE)		1	
NESTED LOOPS		240049	276533
NESTED LOOPS		240049	276533
NESTED LOOPS		60000	96533
HASH JOIN		6000	533
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='Canada'			
TABLE ACCESS (FULL)	CUSTOMER	150000	526
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
INDEX (RANGE SCAN)	IX_CUST_ORDE...	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			



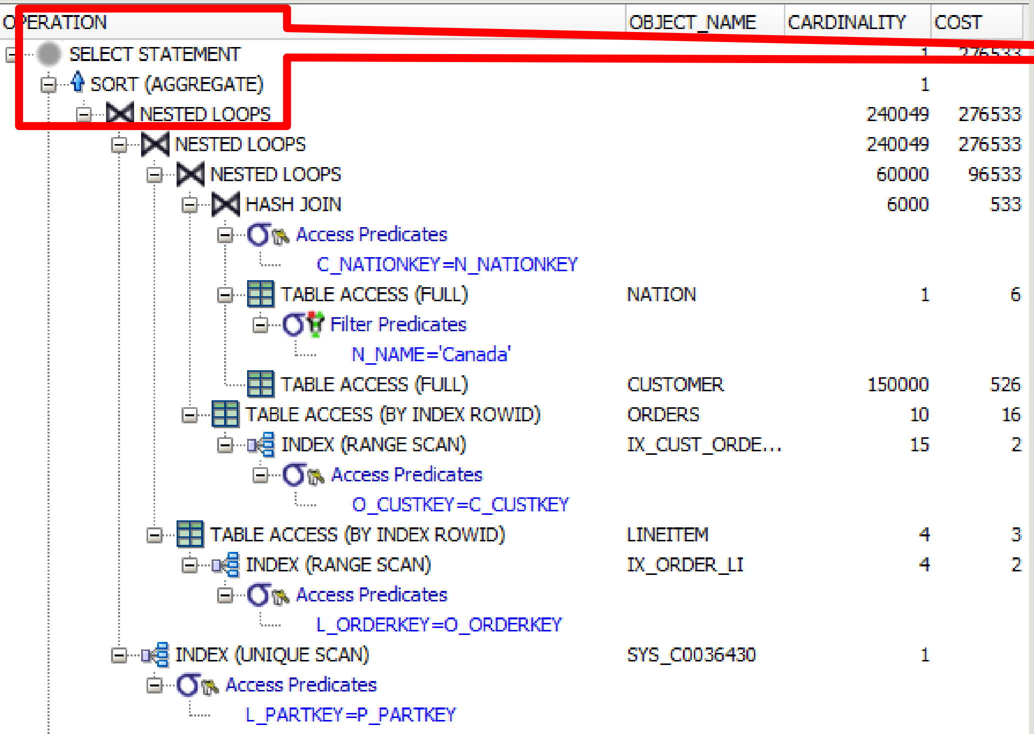
Exercise 5 (see 2)

Draw the execution tree proposed by ORACLE for the following queries:

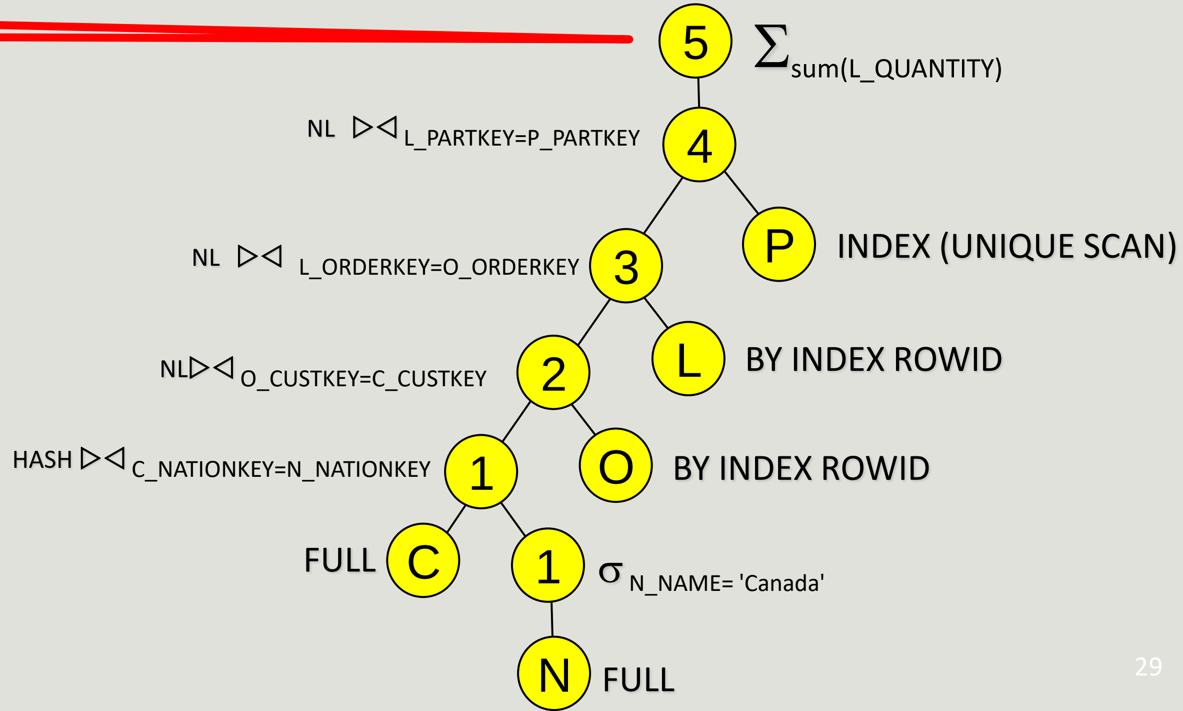
```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```

ORACLE accesses the index but not the Part table. Why?

Actually there would be no need, why?



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	276533
SORT (AGGREGATE)		1	
NESTED LOOPS		240049	276533
NESTED LOOPS		240049	276533
NESTED LOOPS		60000	96533
HASH JOIN		6000	533
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='Canada'			
TABLE ACCESS (FULL)	CUSTOMER	150000	526
TABLE ACCESS (BY INDEX ROWID)	ORDERS	10	16
INDEX (RANGE SCAN)	IX_CUST_ORDE...	15	2
Access Predicates			
O_CUSTKEY=C_CUSTKEY			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			

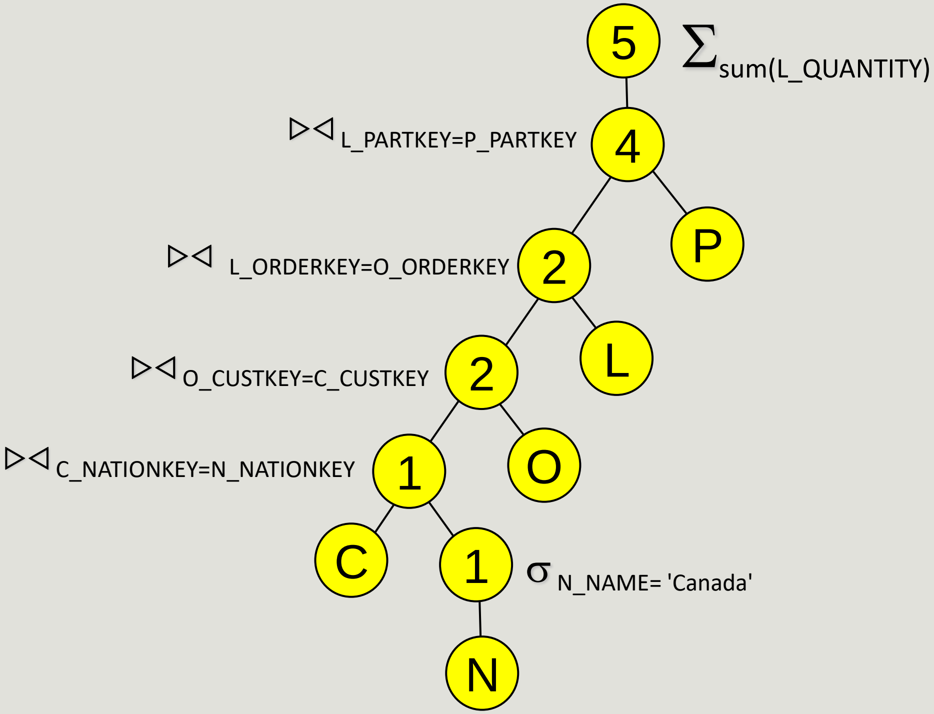


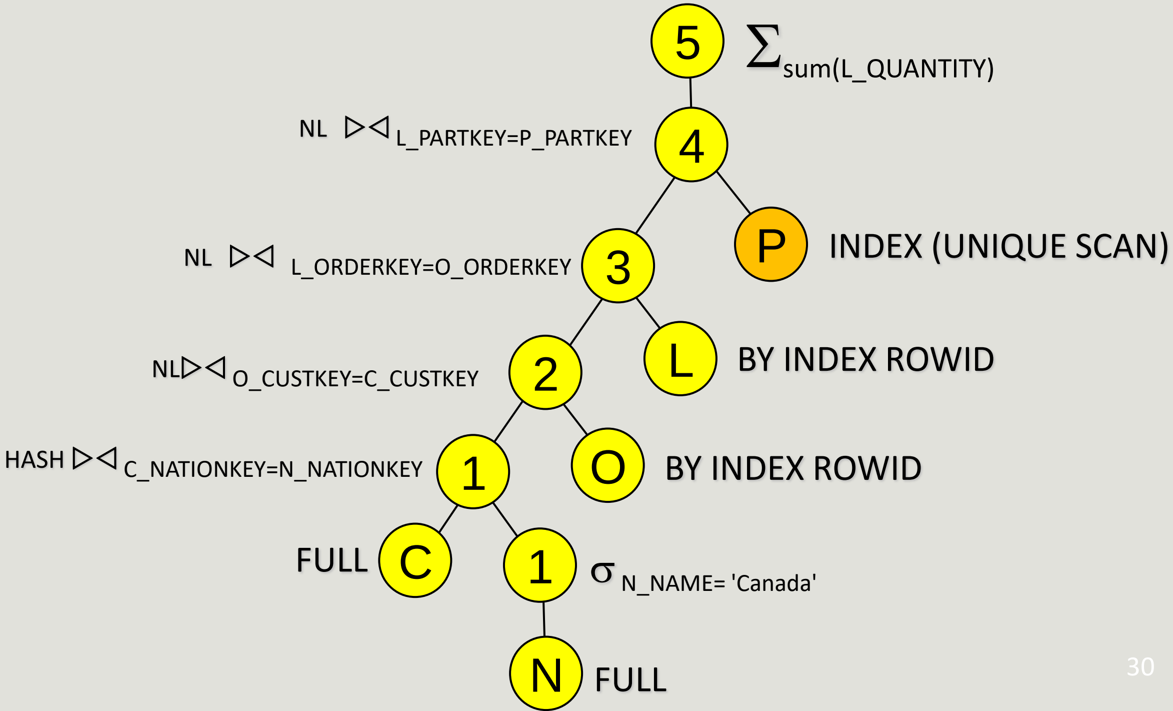
5
$\Sigma_{\text{sum(L\_QUANTITY)}}$
NL $\triangleright\triangleleft$ L_PARTKEY=P_PARTKEY
4
P
INDEX (UNIQUE SCAN)
NL $\triangleright\triangleleft$ L_ORDERKEY=O_ORDERKEY
3
L
BY INDEX ROWID
NL$\triangleright\triangleleft$ O_CUSTKEY=C_CUSTKEY
2
O
BY INDEX ROWID
HASH $\triangleright\triangleleft$ C_NATIONKEY=N_NATIONKEY
1
FULL
C
1
σ N_NAME= 'Canada'
N
FULL

Heuristic plan (see 2) vs Oracle plan

Draw the execution tree proposed by ORACLE for the following queries:

```
select  sum(L_QUANTITY)
from    TPCD.LINEITEM,TPCD.ORDERS,TPCD.PART, TPCD.CUSTOMER, TPCD.NATION
where   L_ORDERKEY=O_ORDERKEY and O_CUSTKEY=C_CUSTKEY and
        C_NATIONKEY=N_NATIONKEY and L_PARTKEY=P_PARTKEY and N_NAME= 'Canada'
```





Exercise 6

Draw the execution tree proposed by ORACLE for the following queries:

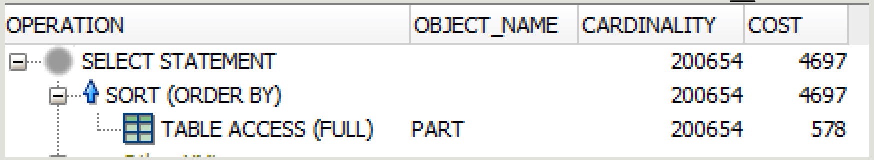
```
select * from PART ORDER BY P_NAME;
```

```
select * from PART ORDER BY P_NAME;
```

Exercise 6

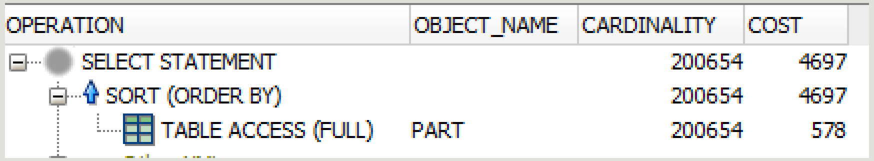
Draw the execution tree proposed by ORACLE for the following queries:

```
select * from PART ORDER BY P_NAME;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		200654	4697
SORT (ORDER BY)		200654	4697
TABLE ACCESS (FULL)	PART	200654	578

```
select * from PART ORDER BY P_PARTKEY;
```

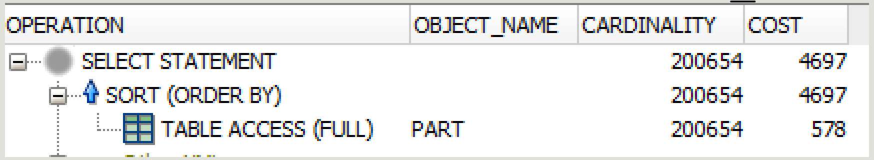


OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		200654	4697
SORT (ORDER BY)		200654	4697
TABLE ACCESS (FULL)	PART	200654	578

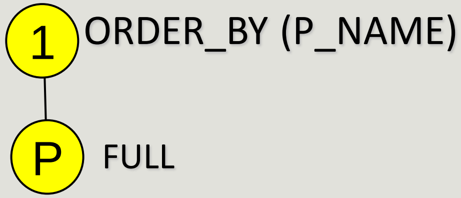
Exercise 6

Draw the execution tree proposed by ORACLE for the following queries:

```
select * from PART ORDER BY P_NAME;
```



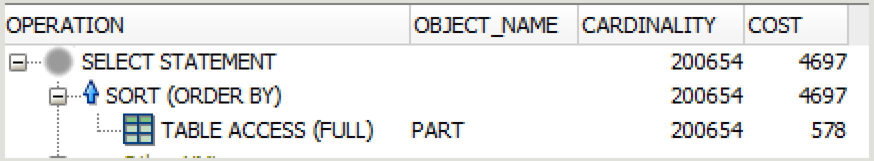
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		200654	4697
SORT (ORDER BY)		200654	4697
TABLE ACCESS (FULL)	PART	200654	578



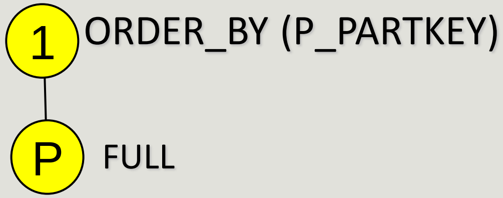
1 ORDER_BY (P_NAME)

P FULL

```
select * from PART ORDER BY P_PARTKEY;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		200654	4697
SORT (ORDER BY)		200654	4697
TABLE ACCESS (FULL)	PART	200654	578



1 ORDER_BY (P_PARTKEY)

P FULL

- Grouping and sorting use the same operator as SORT. The plan indicates the use
- The sort is not really necessary in the second case, why? Because the table is already sorted on P_PARTKEY the key
- In this case **we do not** include the cost in the solution

Exercise 7

After drawing the execution tree of the optimizer for the query:

```
select  o_clerk, p_type, sum(l_quantity),  avg(l_discount)
from    TPCD.lineitem, TPCD.orders, TPCD.part, TPCD.supplier, TPCD.nation
where   l_partkey = p_partkey and l_orderkey = o_orderkey
        and l_suppkey = s_suppkey
        and s_nationkey = n_nationkey
        and o_clerk = 'Clerk#000000955'
        and n_name = 'KENYA'
group   by o_clerk, p_type;
```

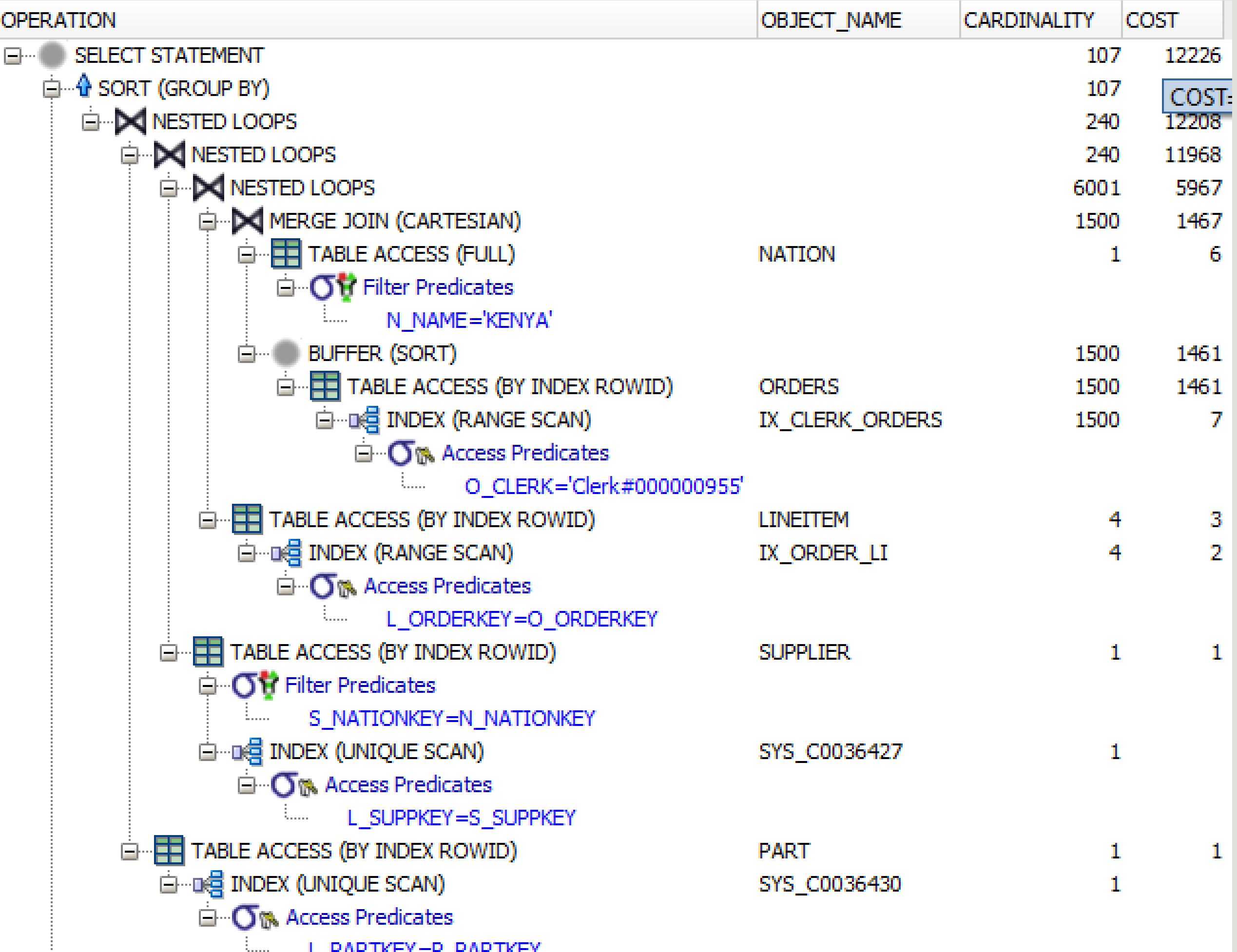
verify:

- How and why the execution tree changes when the o_clerk condition is removed
- How and why the execution tree changes when the conidion on o_clerk is relaxed as follows:

```
o_clerk > 'Clerk#000000955'
```

Exercise 7

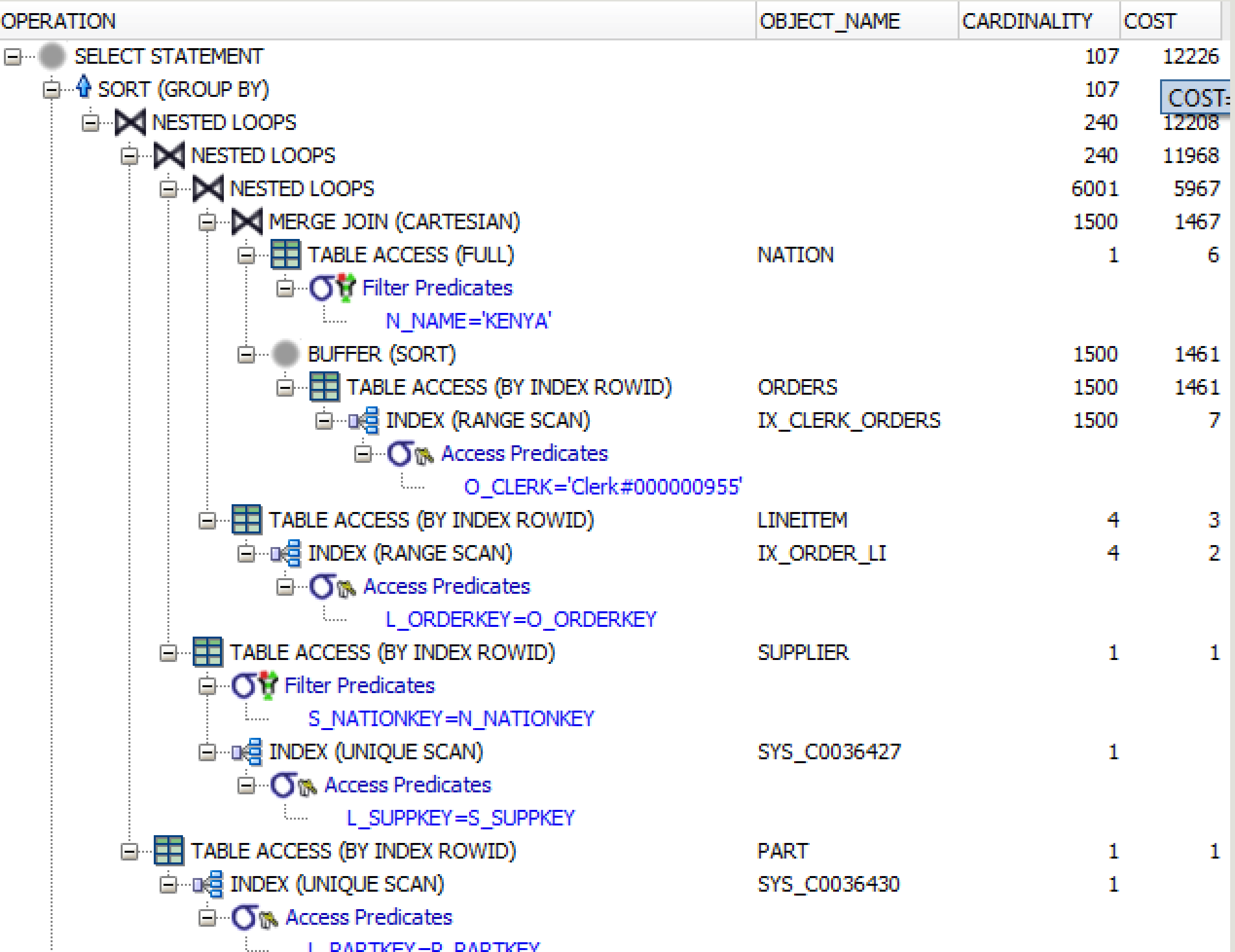
Full query



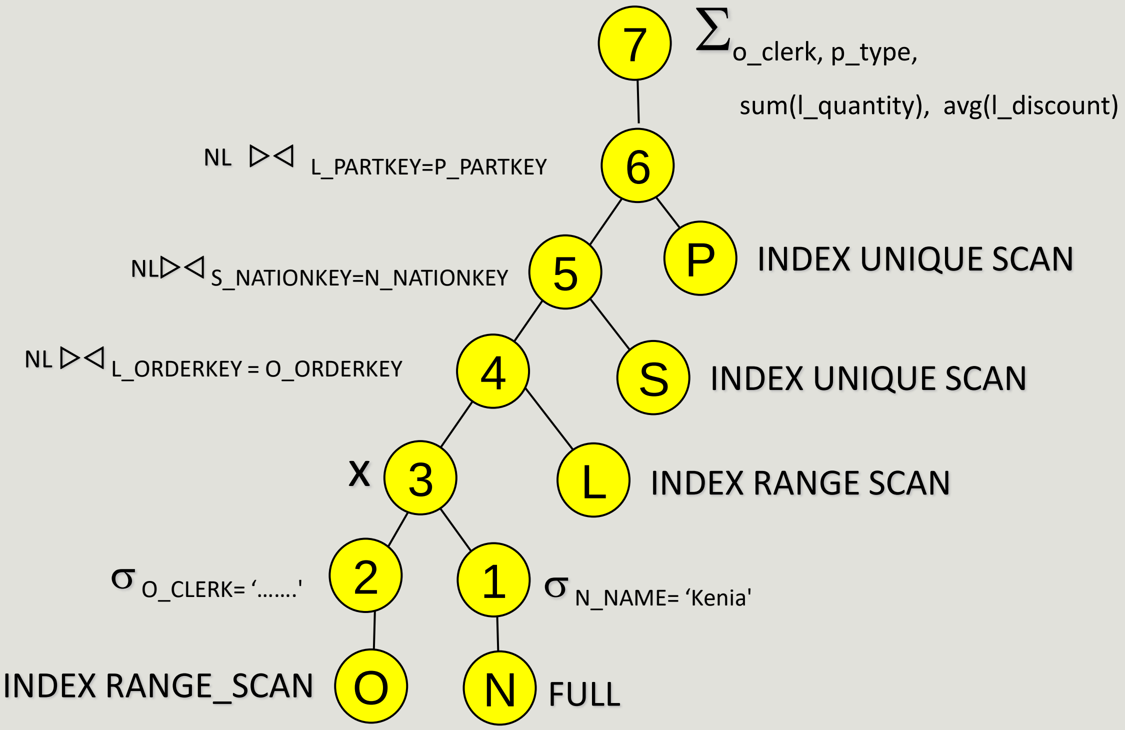
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		107	12226
SORT (GROUP BY)		107	COST=
NESTED LOOPS		240	12208
NESTED LOOPS		240	11968
NESTED LOOPS		6001	5967
MERGE JOIN (CARTESIAN)		1500	1467
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='KENYA'			
BUFFER (SORT)		1500	1461
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1500	1461
INDEX (RANGE SCAN)	IX_CLERK_ORDERS	1500	7
Access Predicates			
O_CLERK='Clerk#000000955'			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
TABLE ACCESS (BY INDEX ROWID)	SUPPLIER	1	1
Filter Predicates			
S_NATIONKEY=N_NATIONKEY			
INDEX (UNIQUE SCAN)	SYS_C0036427	1	
Access Predicates			
L_SUPPKEY=S_SUPPKEY			
TABLE ACCESS (BY INDEX ROWID)	PART	1	1
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			

Exercise 7

Full query



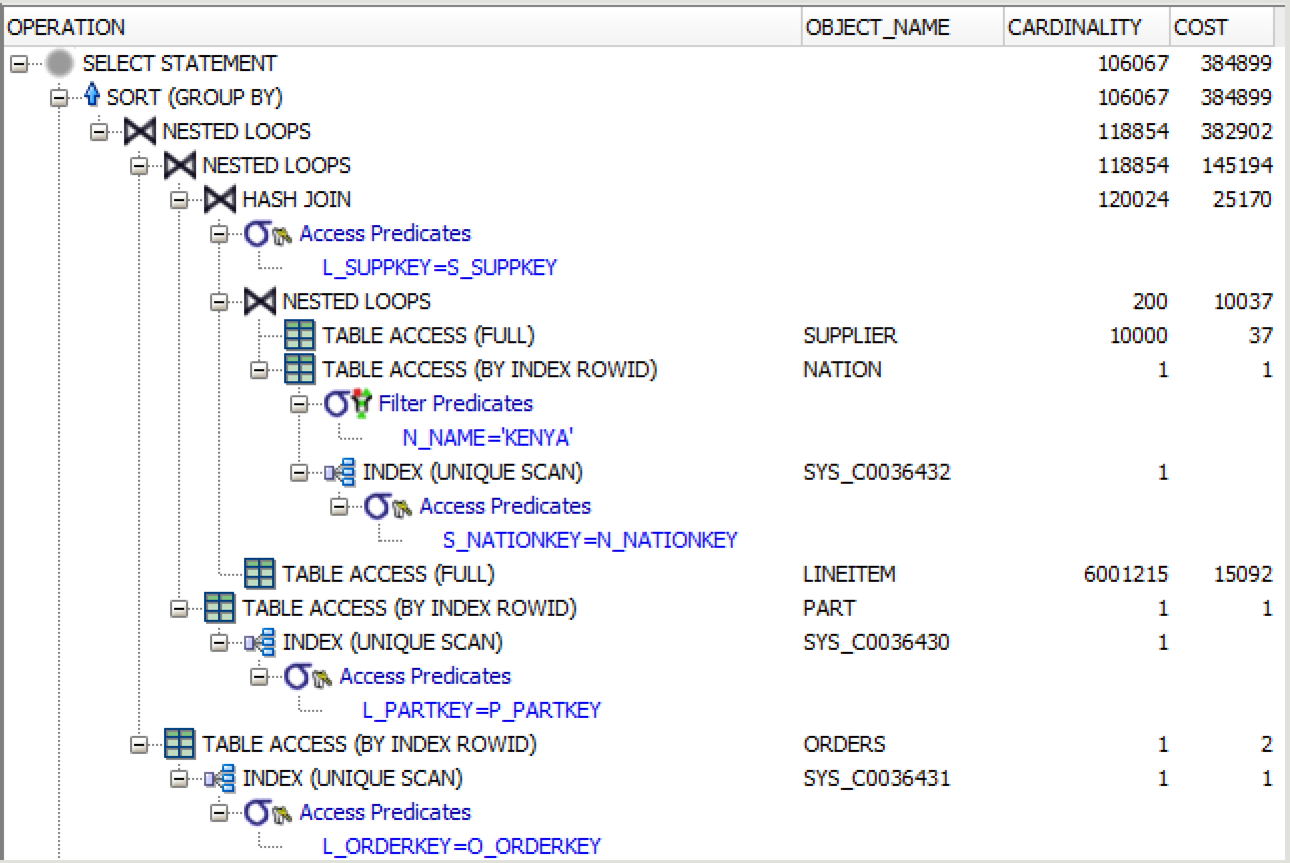
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		107	12226
SORT (GROUP BY)		107	COST=
NESTED LOOPS		240	12208
NESTED LOOPS		240	11968
NESTED LOOPS		6001	5967
MERGE JOIN (CARTESIAN)		1500	1467
TABLE ACCESS (FULL)	NATION	1	6
Filter Predicates			
N_NAME='KENYA'			
BUFFER (SORT)		1500	1461
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1500	1461
INDEX (RANGE SCAN)	IX_CLERK_ORDERS	1500	7
Access Predicates			
O_CLERK='Clerk#000000955'			
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
TABLE ACCESS (BY INDEX ROWID)	SUPPLIER	1	1
Filter Predicates			
S_NATIONKEY=N_NATIONKEY			
INDEX (UNIQUE SCAN)	SYS_C0036427	1	
Access Predicates			
L_SUPPKEY=S_SUPPKEY			
TABLE ACCESS (BY INDEX ROWID)	PART	1	1
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			



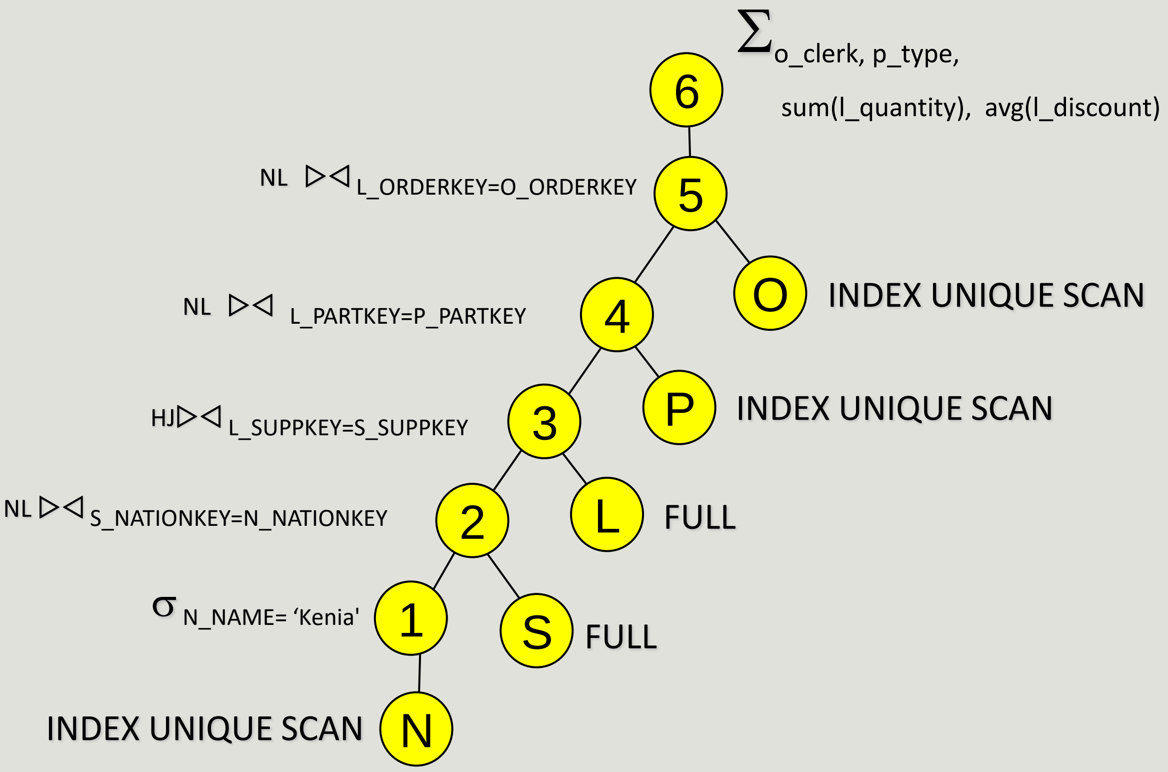
7
$\Sigma_{o\_clerk, p\_type,}$
sum(l_quantity), avg(l_discount)
6
NL ▷◁ L_PARTKEY=P_PARTKEY
P
INDEX UNIQUE SCAN
5
NL▷◁ S_NATIONKEY=N_NATIONKEY
S
INDEX UNIQUE SCAN
4
NL ▷◁ L_ORDERKEY = O_ORDERKEY
L
INDEX RANGE SCAN
X
3
2
σ O_CLERK= '…….'
1
σ N_NAME= 'Kenia'
O
INDEX RANGE_SCAN
N
FULL

Exercise 7

without the condition on `o_clerk`



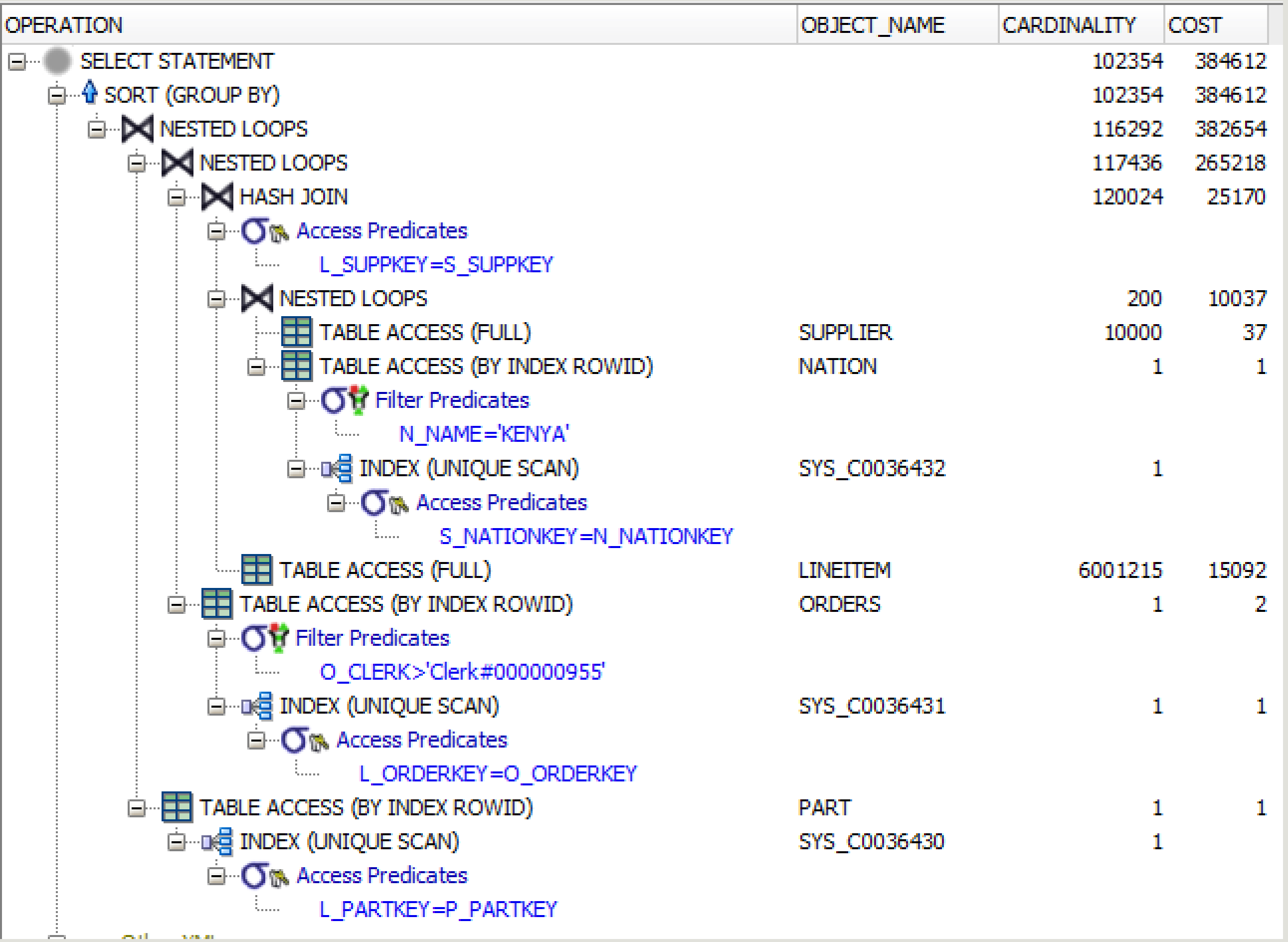
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		106067	384899
SORT (GROUP BY)		106067	384899
NESTED LOOPS		118854	382902
NESTED LOOPS		118854	145194
HASH JOIN		120024	25170
Access Predicates			
L_SUPPKEY=S_SUPPKEY			
NESTED LOOPS		200	10037
TABLE ACCESS (FULL)	SUPPLIER	10000	37
TABLE ACCESS (BY INDEX ROWID)	NATION	1	1
Filter Predicates			
N_NAME='KENYA'			
INDEX (UNIQUE SCAN)	SYS_C0036432	1	
Access Predicates			
S_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	LINEITEM	6001215	15092
TABLE ACCESS (BY INDEX ROWID)	PART	1	1
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1	2
INDEX (UNIQUE SCAN)	SYS_C0036431	1	1
Access Predicates			
L_ORDERKEY=O_ORDERKEY			



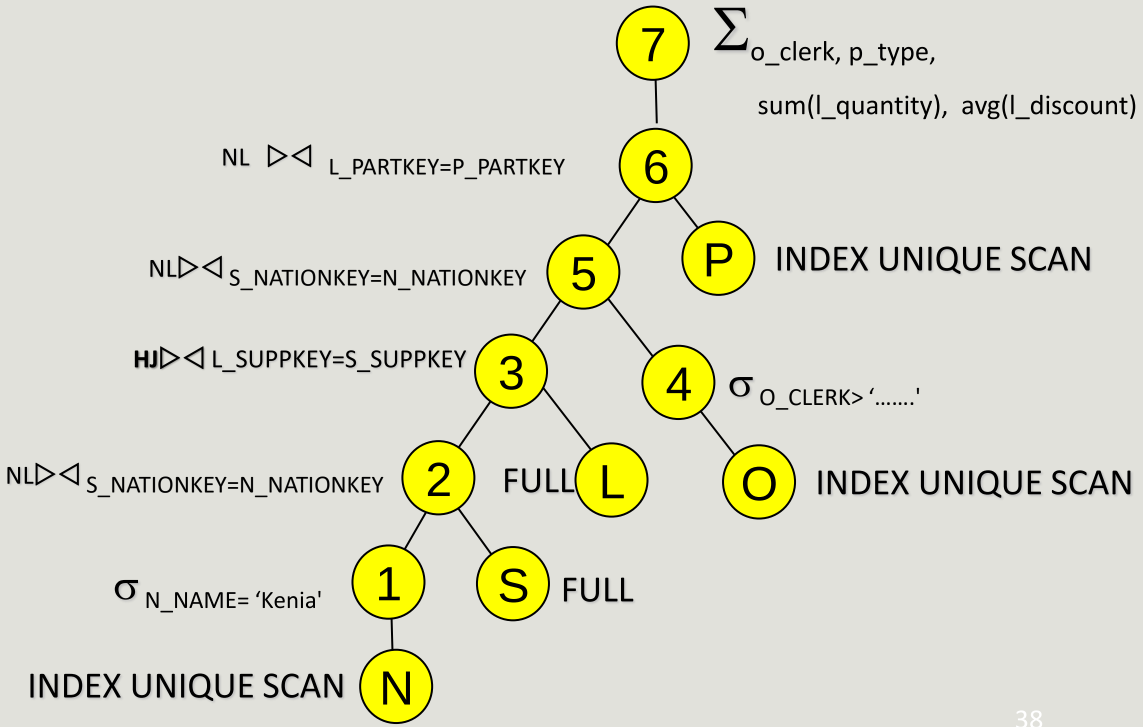
$\Sigma_{o\_clerk,\ p\_type,}$ sum(l_quantity), avg(l_discount)
6
NL ▷◁ L_ORDERKEY=O_ORDERKEY
5
O
INDEX UNIQUE SCAN
NL ▷◁ L_PARTKEY=P_PARTKEY
4
P
INDEX UNIQUE SCAN
HJ▷◁ L_SUPPKEY=S_SUPPKEY
3
L
FULL
NL ▷◁ S_NATIONKEY=N_NATIONKEY
2
S
FULL
σ N_NAME= 'Kenia'
1
INDEX UNIQUE SCAN
N

Exercise 7

With relaxed condition `o_clerk >'Clerk#000000955'`



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		102354	384612
SORT (GROUP BY)		102354	384612
NESTED LOOPS		116292	382654
NESTED LOOPS		117436	265218
HASH JOIN		120024	25170
Access Predicates			
L_SUPPKEY=S_SUPPKEY			
NESTED LOOPS		200	10037
TABLE ACCESS (FULL)	SUPPLIER	10000	37
TABLE ACCESS (BY INDEX ROWID)	NATION	1	1
Filter Predicates			
N_NAME='KENYA'			
INDEX (UNIQUE SCAN)	SYS_C0036432	1	
Access Predicates			
S_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	LINEITEM	6001215	15092
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1	2
Filter Predicates			
O_CLERK>'Clerk#000000955'			
INDEX (UNIQUE SCAN)	SYS_C0036431	1	1
Access Predicates			
L_ORDERKEY=O_ORDERKEY			
TABLE ACCESS (BY INDEX ROWID)	PART	1	1
INDEX (UNIQUE SCAN)	SYS_C0036430	1	
Access Predicates			
L_PARTKEY=P_PARTKEY			



Execution Cost Computation

ADVANCED DATA BASE

Business Intelligence Group

Matteo Golfarelli

Exercise 8

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select R_NAME, count(*) AS NCUST
from TPCD.REGION, TPCD.NATION, TPCD.CUSTOMER
where R_REGIONKEY=N_REGIONKEY AND N_NATIONKEY=C_NATIONKEY
GROUP BY R_NAME,R_REGIONKEY
ORDER  BY NCUST;
```

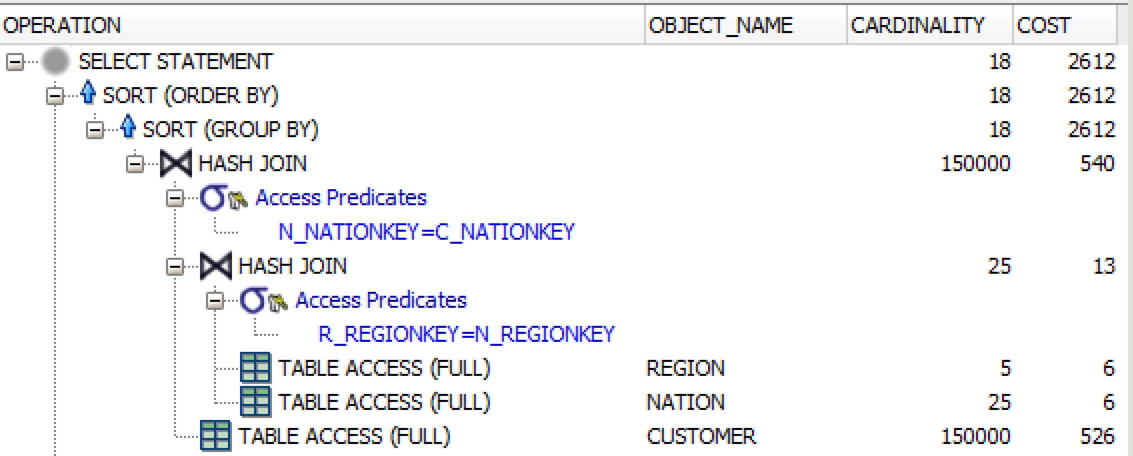
Exercise 8

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select R_NAME, count(*) AS NCUST
from TPCD.REGION, TPCD.NATION, TPCD.CUSTOMER
where R_REGIONKEY=N_REGIONKEY AND N_NATIONKEY=C_NATIONKEY
GROUP BY R_NAME,R_REGIONKEY
ORDER  BY NCUST;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		18	2612
SORT (ORDER BY)		18	2612
SORT (GROUP BY)		18	2612
HASH JOIN		150000	540
Access Predicates			
N_NATIONKEY=C_NATIONKEY			
HASH JOIN		25	13
Access Predicates			
R_REGIONKEY=N_REGIONKEY			
TABLE ACCESS (FULL)	REGION	5	6
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	CUSTOMER	150000	526

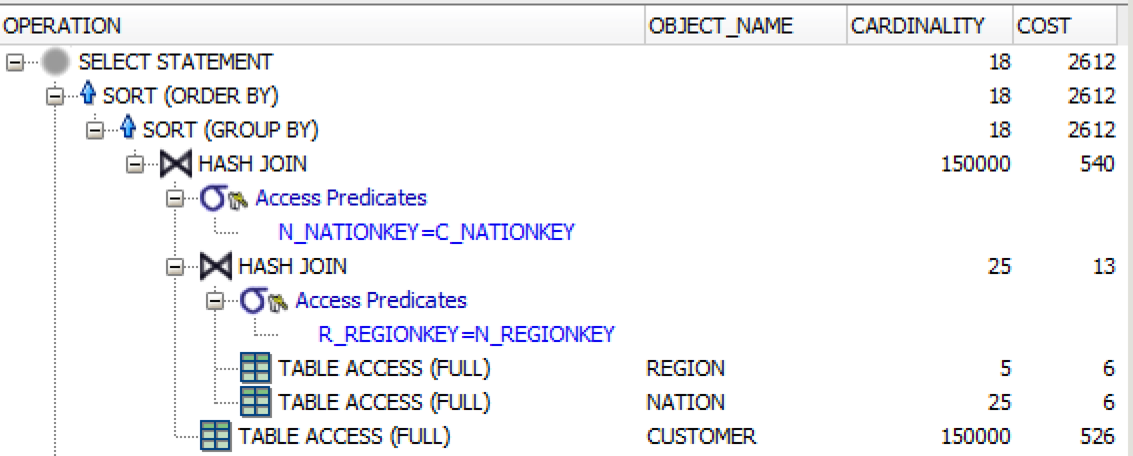
Exercise 8

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

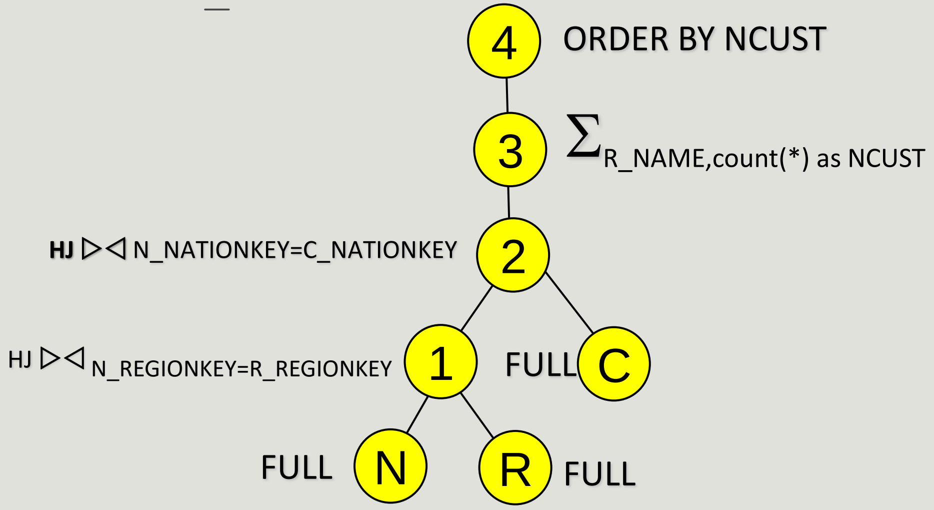
D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select R_NAME, count(*) AS NCUST
from TPCD.REGION, TPCD.NATION, TPCD.CUSTOMER
where R_REGIONKEY=N_REGIONKEY AND N_NATIONKEY=C_NATIONKEY
GROUP BY R_NAME,R_REGIONKEY
ORDER  BY NCUST;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		18	2612
SORT (ORDER BY)		18	2612
SORT (GROUP BY)		18	2612
HASH JOIN		150000	540
Access Predicates			
N_NATIONKEY=C_NATIONKEY			
HASH JOIN		25	13
Access Predicates			
R_REGIONKEY=N_REGIONKEY			
TABLE ACCESS (FULL)	REGION	5	6
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	CUSTOMER	150000	526


4
ORDER BY NCUST
3
$\Sigma_{\text{R\_NAME,count(*) as NCUST}}$
HJ ▷◁ N_NATIONKEY=C_NATIONKEY
2
HJ ▷◁ N_REGIONKEY=R_REGIONKEY
1
FULL
C
FULL
N
R
FULL

Exercise 8

$NP_{REGION} = \lceil 5 \times 114/\ (4096 \times 0{,}69) \rceil = 1$

$NP_{NATION} = \lceil 25 \times 106/\ (4096 \times 0{,}69) \rceil = 1$

$NP_{REGION+NATION} = \lceil 25 \times (114+106)\ /\ (4096 \times 0{,}69) \rceil = 2$

$NP_{CUSTOMER} = \lceil 150.000 \times 159/\ (4096 \times 0{,}69) \rceil = 8.439$

$NP_{REGION+NATION+CUSTOMER} = \lceil 150.000 \times (114+106+159)\ /\ (4096 \times 0{,}69) \rceil = 20.116$

We can use l'hash join since NP_{NATION} e NP_{REGION} are smaller than NB

Hash $Join_{REGION+NATION}$ = **2**

We can use l'hash join since $NP_{REGION+NATION}$ is lower than NB

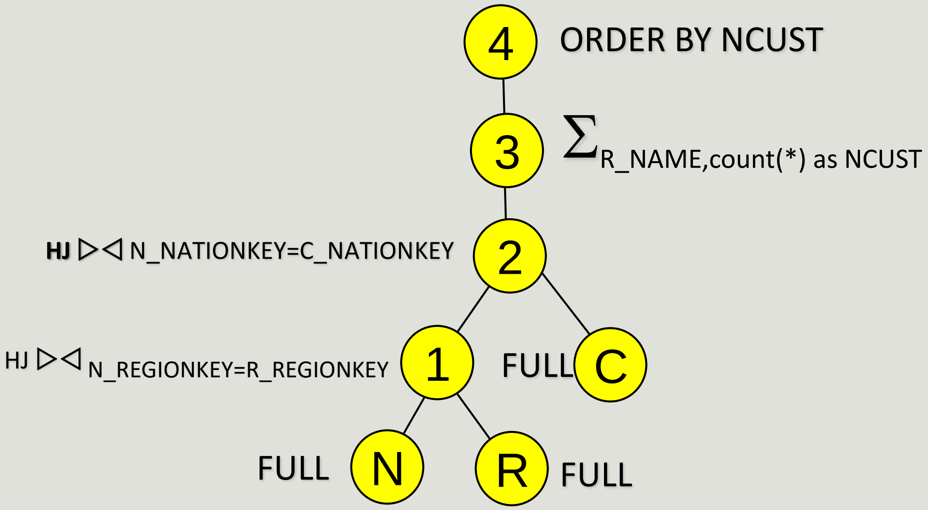
Hash $Join_{(REGION+NATION)+CUSTOMER}$ = 2 + 8.439 = **8.441**

Group by access cost = $2 \times 20.116 \times (\lceil \log_{100} 20.116 \rceil + 1) = 2 \times 20.116 \times (3\ + 1)$ = **160.928**

$NP_{GB} = \lceil 5 \times (\ 4+26)\ \ /\ (4096 \times 0{,}69) \rceil = 1$

Inner sort since $NP_{RESULT} < NB$

Total cost = 2 + 8.441 + 160.928 + 1 = **169.372**


4
ORDER BY NCUST
3
$\Sigma_{R\_NAME,count(*)\ as\ NCUST}$
HJ $\rhd\lhd$ N_NATIONKEY=C_NATIONKEY
2
HJ $\rhd\lhd$ N_REGIONKEY=R_REGIONKEY
1
FULL
C
FULL
N
R
FULL

Exercise 9

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte len(P)=len(K)=4 byte NB = 101 u = 0.69

No projections on intermediate results

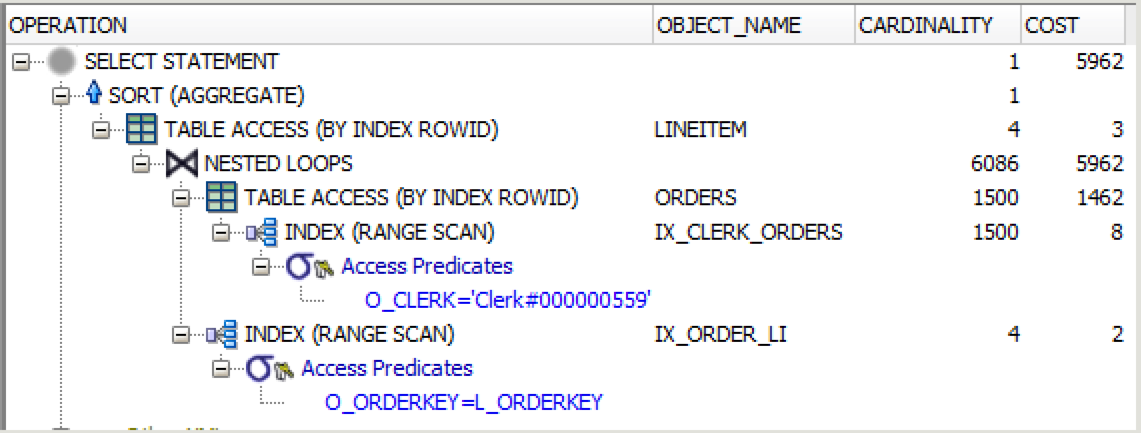
```
select sum(L_EXTENDEDPRICE)
from TPCD.ORDERS, TPCD.LINEITEM
WHERE O_ORDERKEY=L_ORDERKEY
and O_CLERK='Clerk#000000559';
```

Exercise 9

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69   No projections on intermediate results

```
select sum(L_EXTENDEDPRICE)
from TPCD.ORDERS, TPCD.LINEITEM
WHERE O_ORDERKEY=L_ORDERKEY
and O_CLERK='Clerk#000000559';
```



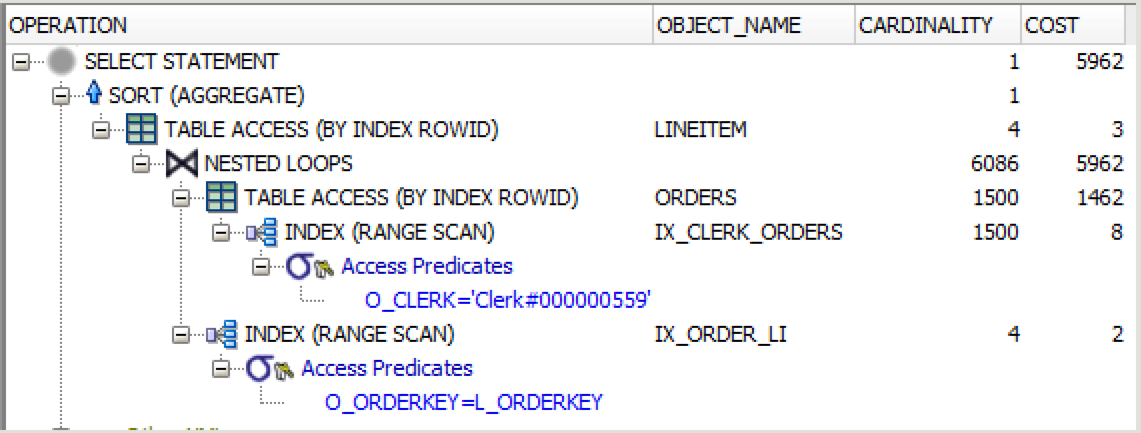
OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	5962
SORT (AGGREGATE)		1	
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
NESTED LOOPS		6086	5962
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1500	1462
INDEX (RANGE SCAN)	IX_CLERK_ORDERS	1500	8
Access Predicates			
O_CLERK='Clerk#000000559'			
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
O_ORDERKEY=L_ORDERKEY			

Exercise 9

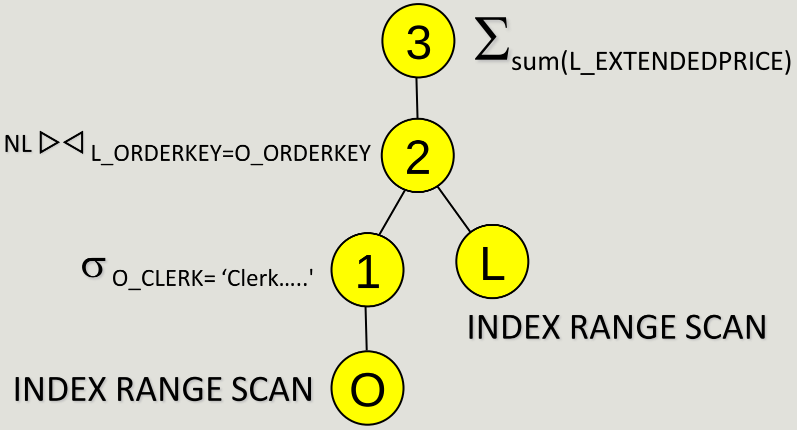
After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69   No projections on intermediate results

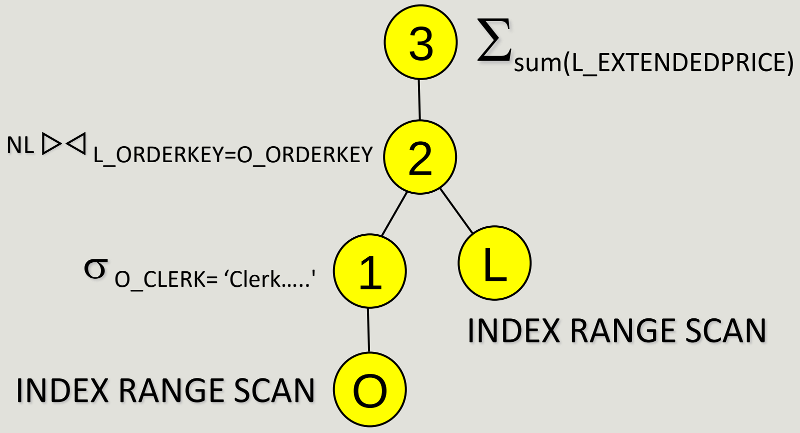
```
select sum(L_EXTENDEDPRICE)
from TPCD.ORDERS, TPCD.LINEITEM
WHERE O_ORDERKEY=L_ORDERKEY
and O_CLERK='Clerk#000000559';
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		1	5962
SORT (AGGREGATE)		1	
TABLE ACCESS (BY INDEX ROWID)	LINEITEM	4	3
NESTED LOOPS		6086	5962
TABLE ACCESS (BY INDEX ROWID)	ORDERS	1500	1462
INDEX (RANGE SCAN)	IX_CLERK_ORDERS	1500	8
Access Predicates			
O_CLERK='Clerk#000000559'			
INDEX (RANGE SCAN)	IX_ORDER_LI	4	2
Access Predicates			
O_ORDERKEY=L_ORDERKEY			


3
$\Sigma_{sum(L\_EXTENDEDPRICE)}$
NL $\rhd\lhd_{L\_ORDERKEY=O\_ORDERKEY}$
2
$\sigma_{O\_CLERK= 'Clerk.....'}$
1
L
INDEX RANGE SCAN
INDEX RANGE SCAN
O

Exercise 9


3
$\Sigma_{sum(L\_EXTENDEDPRICE)}$
NL $\bowtie_{L\_ORDERKEY=O\_ORDERKEY}$
2
$\sigma_{O\_CLERK= 'Clerk.....'}$
1
L
INDEX RANGE SCAN
INDEX RANGE SCAN
O

$NP_{ORDERS} = \lceil 1.500.000 \times 106 / (4096 \times 0{,}69) \rceil = 56.259$

$NP_{LINEITEMS} = \lceil 6.001.215 \times 113 / (4.096 \times 0{,}69) \rceil = 239.944$

$NL_{O\_CLERK} = \lceil (1.000 \times 4 + 4 \times 1.500.000) / (4096 \times 0{,}69) \rceil = 2.125$

$|O\_CLERK| = 1.000$

$Sel(O\_CLERK='...') = 1/1000$

$ET_{ORDERS} = 1.500.000 / 1.000 = 1.500$

$h-1 = BLEVEL_{IX\_CLERK\_ORDERS} = 2$

Unclustered access ORDERS $= 2 + \lceil 1/1.000 \times 2.125 \rceil + 1 \times \Phi(1.500, 56.259) = 2 + 3 + 1.481 = 1.486$

$NL_{L\_ORDERKEY} = \lceil (1.500.000 \times 4 + 4 \times 6.001.215) / (4096 \times 0{,}69) \rceil = 10.617$

$h-1 = BLEVEL_{IX\_ORDER\_LI} = 2$

Clustered access to LINEITEM $= 2 + \lceil 1/1.500.000 \times 10.617 \rceil + 1 \times \lceil 1/1.500.000 \times 239.944 \rceil = 2 + 1 + 1 = 4$

NL $Join_{LINEITEM+ORDERS} = 1.486 + 1.500 \times 4 =$ **7.486**

$NP_{LINEITEM+ORDERS} = \lceil (106+113) \times (6.001.215/1.000) / (4096 \times 0.69) \rceil = 466$

Sort(LINEITEM+ORDERS) $= 2 \times 466 \times (\lceil \log_{100} 466 \rceil + 1) = 2 \times 466 \times (2 + 1) =$ **2.796**

Total cost = 7.486 + 2.796 = **10.282**

Exercise 10

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69   No projections on intermediate results

```
SELECT sum(PS_SUPPLYCOST)
FROM TPCD.PART,TPCD.PARTSUPP
WHERE P_PARTKEY=PS_PARTKEY and P_TYPE='SMALL BURNISHED STEEL';
```

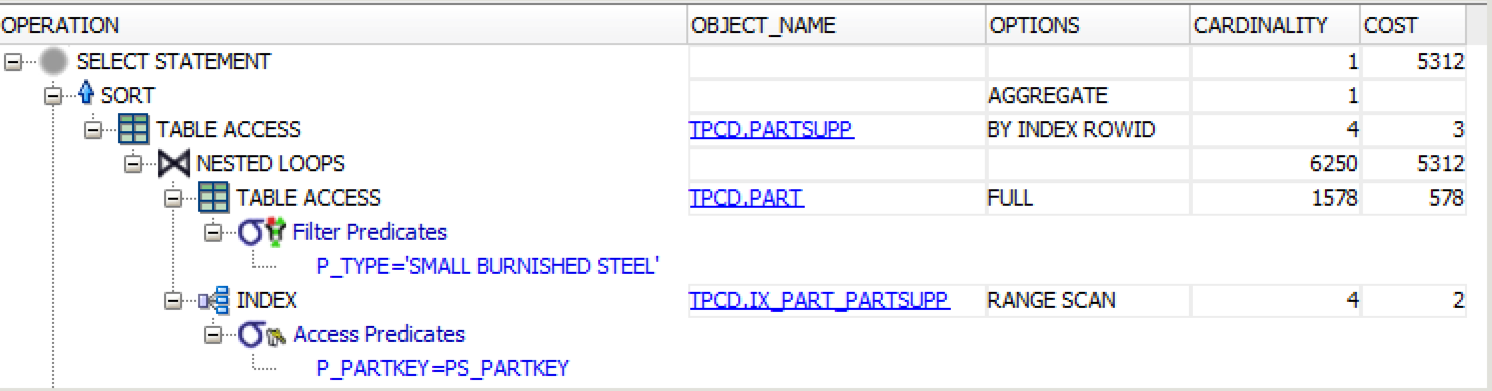
Exercise 10

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte len(P)=len(K)=4 byte NB = 101 u = 0.69

No projections on
intermediate results

```
SELECT sum(PS_SUPPLYCOST)
FROM TPCD.PART,TPCD.PARTSUPP
WHERE P_PARTKEY=PS_PARTKEY and P_TYPE='SMALL BURNISHED STEEL';
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	5312
SORT		AGGREGATE	1	
TABLE ACCESS	TPCD.PARTSUPP	BY INDEX ROWID	4	3
NESTED LOOPS			6250	5312
TABLE ACCESS	TPCD.PART	FULL	1578	578
Filter Predicates				
P_TYPE='SMALL BURNISHED STEEL'				
INDEX	TPCD.IX_PART_PARTSUPP	RANGE SCAN	4	2
Access Predicates				
P_PARTKEY=PS_PARTKEY				

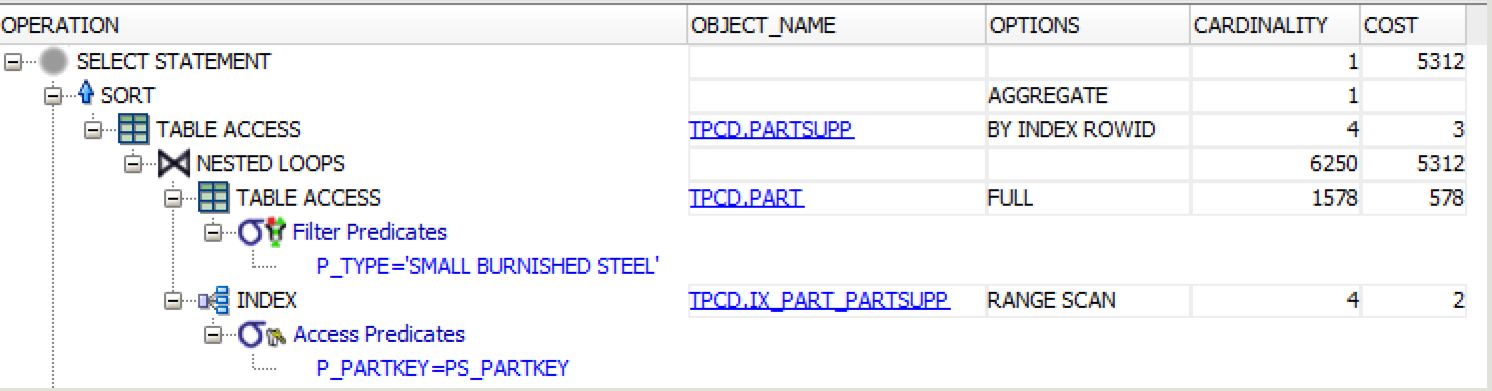
Exercise 10

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

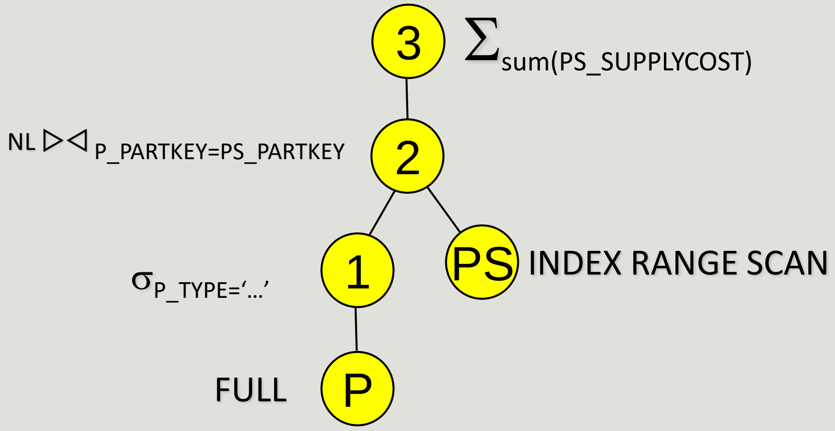
D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

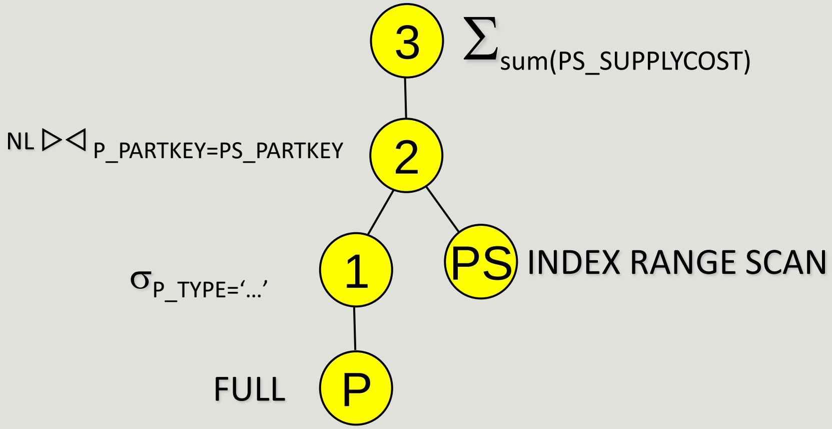
```
SELECT sum(PS_SUPPLYCOST)
FROM TPCD.PART,TPCD.PARTSUPP
WHERE P_PARTKEY=PS_PARTKEY and P_TYPE='SMALL BURNISHED STEEL';
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			1	5312
SORT		AGGREGATE	1	
TABLE ACCESS	TPCD.PARTSUPP	BY INDEX ROWID	4	3
NESTED LOOPS			6250	5312
TABLE ACCESS	TPCD.PART	FULL	1578	578
Filter Predicates				
P_TYPE='SMALL BURNISHED STEEL'				
INDEX	TPCD.IX_PART_PARTSUPP	RANGE SCAN	4	2
Access Predicates				
P_PARTKEY=PS_PARTKEY				


3
$\Sigma_{sum(PS\_SUPPLYCOST)}$
NL $\triangleright\triangleleft_{P\_PARTKEY=PS\_PARTKEY}$
2
$\sigma_{P\_TYPE='...'}$
1
PS
INDEX RANGE SCAN
FULL
P

Exercise 10



$\Sigma_{sum(PS\_SUPPLYCOST)}$

3

NL $\triangleright\triangleleft_{P\_PARTKEY=PS\_PARTKEY}$

2

$\sigma_{P\_TYPE='...'}$

1

PS INDEX RANGE SCAN

FULL

P

$NP_{PART} = \lceil 200.000 \times 131/(4096 \times 0{,}69) \rceil = 9.271$

Sel(P_TYPE='…')=1/150

$ET_{PART} = \lceil 200.000 \,/\, 150 \rceil = 1.334$

$NP_{PARTSUPP} = \lceil 800.000 \times 143/(4096 \times 0{,}69) \rceil = 40.478$

$NL_{PS\_PARTKEY} = \lceil 200.000 \times 4 + 800.000 \times 4) \,/\, (4096 \times 0{,}69) \rceil = 1.416$

Clustered access to PARTSUPP= $2 + \lceil 1/\, 200.000 \times 1.416 \rceil + \lceil 1/200000 \times 40.478 \rceil = 2+1+1=$**4**

NL Join$_{LINEITEM+ORDERS}$ $= 9.271 + 1.334 \times 4 =$ **14.607**

$NT_{\ PART+PARTSUPP\ =} \lceil 800.000 \times 1/150 \rceil = 5.334$

$NP_{\ PART+PARTSUPP} = \lceil 5.334 \times (131+143)/(4096 \times 0{,}69) \rceil = 518$

Group by (PART + PARTSUPP) $= 2 \times 518 \times (\lceil \log_{100} 518 \rceil + 1) = 2 \times 518 \times (2 + 1) =$ **3.108**

Total cost = **14.607 + 3.108 = 17.715**

Exercise 11

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select P_TYPE,SUM(L_QUANTITY)
from TPCD.LINEITEM, TPCD.PART
where L_PARTKEY=P_PARTKEY and P_BRAND= 'Brand#54'
group by P_TYPE
having COUNT(*) > 5;
```

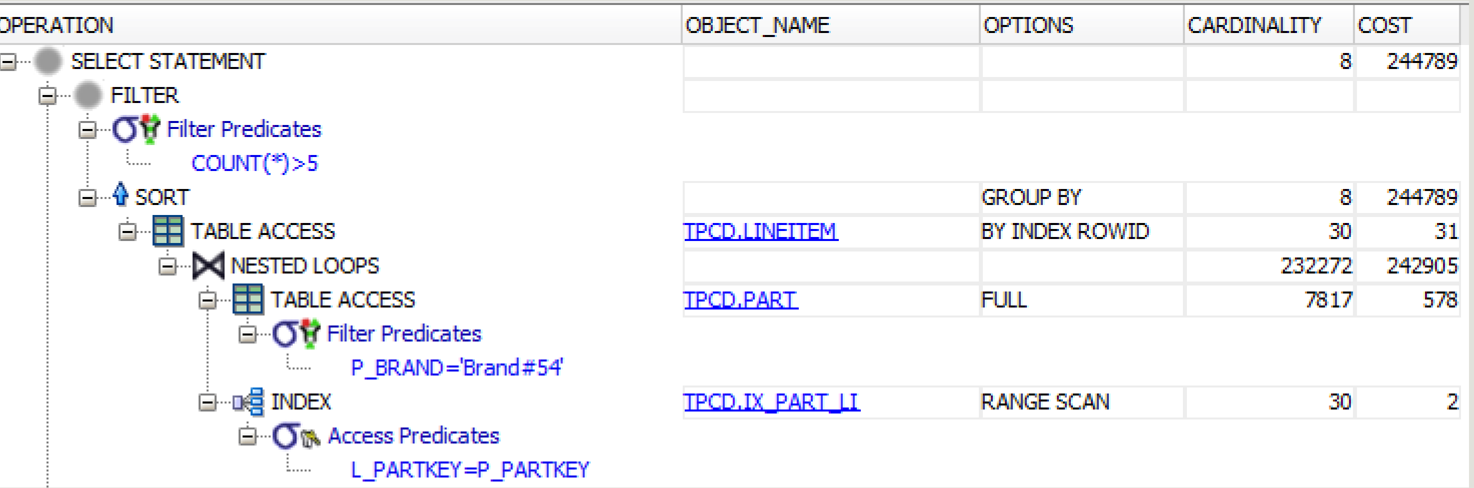
Exercise 11

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select P_TYPE,SUM(L_QUANTITY)
from TPCD.LINEITEM, TPCD.PART
where L_PARTKEY=P_PARTKEY and P_BRAND= 'Brand#54'
group by P_TYPE
having COUNT(*) > 5;
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			8	244789
FILTER				
Filter Predicates				
COUNT(*)>5				
SORT		GROUP BY	8	244789
TABLE ACCESS	TPCD.LINEITEM	BY INDEX ROWID	30	31
NESTED LOOPS			232272	242905
TABLE ACCESS	TPCD.PART	FULL	7817	578
Filter Predicates				
P_BRAND='Brand#54'				
INDEX	TPCD.IX_PART_LI	RANGE SCAN	30	2
Access Predicates				
L_PARTKEY=P_PARTKEY				

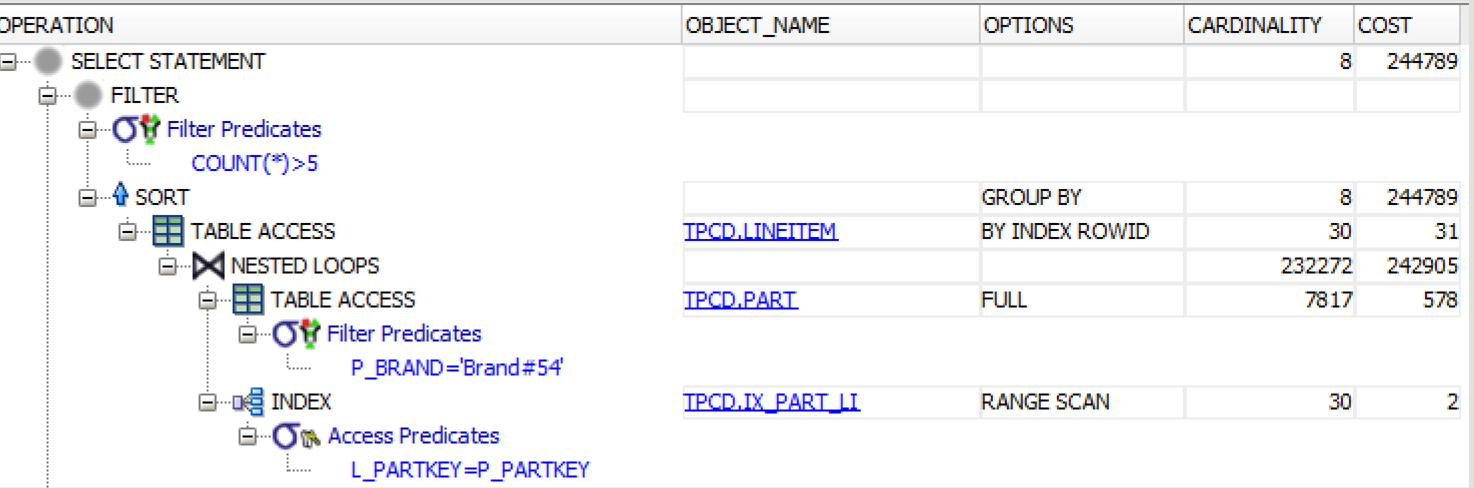
Exercise 11

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

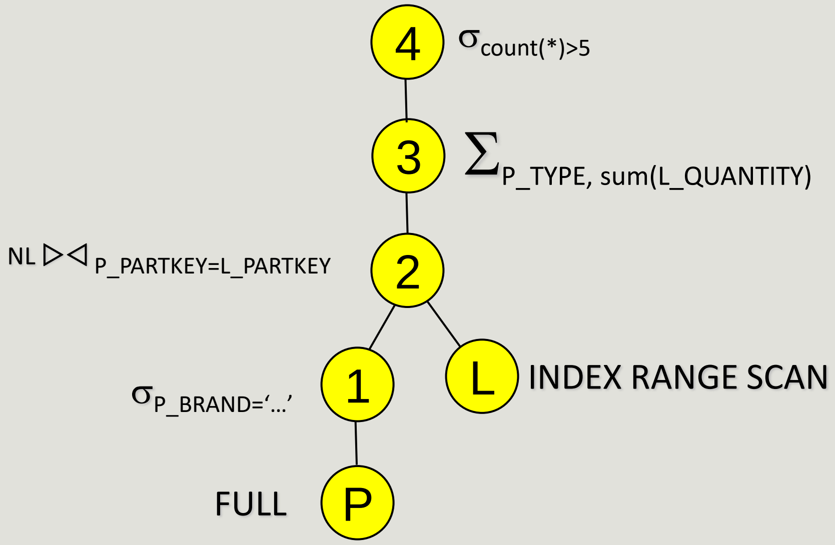
D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

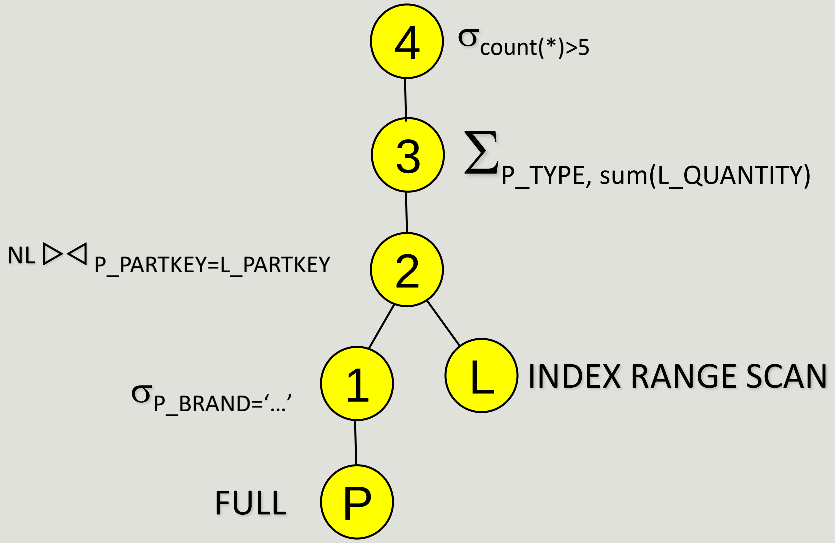
```
select P_TYPE,SUM(L_QUANTITY)
from TPCD.LINEITEM, TPCD.PART
where L_PARTKEY=P_PARTKEY and P_BRAND= 'Brand#54'
group by P_TYPE
having COUNT(*) > 5;
```



OPERATION	OBJECT_NAME	OPTIONS	CARDINALITY	COST
SELECT STATEMENT			8	244789
FILTER				
Filter Predicates				
COUNT(*)>5				
SORT		GROUP BY	8	244789
TABLE ACCESS	TPCD.LINEITEM	BY INDEX ROWID	30	31
NESTED LOOPS			232272	242905
TABLE ACCESS	TPCD.PART	FULL	7817	578
Filter Predicates				
P_BRAND='Brand#54'				
INDEX	TPCD.IX_PART_LI	RANGE SCAN	30	2
Access Predicates				
L_PARTKEY=P_PARTKEY				


4
$\sigma_{count(*)>5}$
3
$\Sigma_{P\_TYPE,\ sum(L\_QUANTITY)}$
NL $\triangleright\triangleleft_{P\_PARTKEY=L\_PARTKEY}$
2
$\sigma_{P\_BRAND='...'}$
1
L
INDEX RANGE SCAN
FULL
P

Exercise 11


4
$\sigma_{count(*)>5}$
3
$\Sigma_{P\_TYPE,\ sum(L\_QUANTITY)}$
NL $\triangleright\triangleleft_{P\_PARTKEY=L\_PARTKEY}$
2
$\sigma_{P\_BRAND='...'}$
1
L
INDEX RANGE SCAN
FULL
P

$NP_{PART} = \lceil 200.000 \times 131/(4096 \times 0{,}69) \rceil = 9.271$

Sel(P_BRAND='…')=1/25

$ET_{PART} = \lceil 200.000 \times 1/25 \rceil = 8.000$

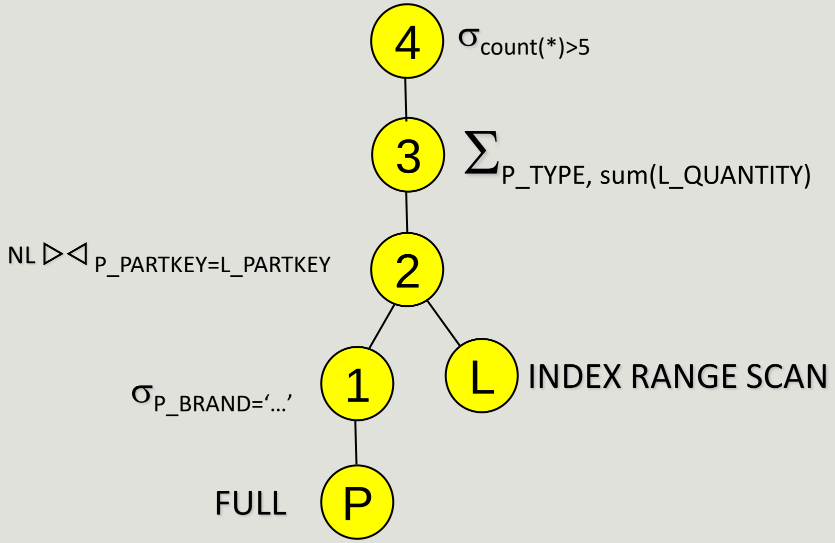
$NP_{LINEITEM} = \lceil 6.001.215 \times 113 / (4.096 \times 0{,}69) \rceil = 239.944$

$NL_{P\text{-}LI} = \lceil 200.000 \times 4 + 6.001.215 \times 4) / (4096 \times 0{,}69 \rceil = 8.777$

Uncl. access to LI = $2 + \lceil 8.777/200.000 \rceil + \Phi(6.001.215/200.000,\ 239.944) = 2 + 1 + \Phi(31,\ 239.944) = 2+1+31 = 34$

Cost NL $Join_{P\text{-}LI}$ 9.271+8.000 × 34=**257.271**

Exercise 11


4
$\sigma_{count(*)>5}$
3
$\Sigma_{P\_TYPE,\ sum(L\_QUANTITY)}$
NL $\bowtie_{P\_PARTKEY=L\_PARTKEY}$
2
$\sigma_{P\_BRAND='...'}$
1
L
INDEX RANGE SCAN
FULL
P

$ETL_{LI\text{-}P}$= 8.000 × 6.001.215 / 200.000 = 8000 × 30 = 240.049

Alternatively $ETL_{LI\text{-}P}$ = 6.001.215 / 25 = 240.049

$NP_{P\text{-}LI} = \lceil 240.049 \times (113+131) / (4.096 \times 0{,}69) \rceil = 20.725$

Sort $GB_{P\text{-}LI} = 2 \times 20.725 \times (\lceil \log_{100} 20.725 \rceil + 1) = 20.725 \times 8 =$ **165.800**

The number of tuples after the group is estimated through the Cardenas formula which estimates how the 240,049 tuples are grouped with respect to the values of the parts to which they refer

$ET_{GB\text{-}P\_TYPE} = \Phi(240.049, 150) = 150$

$NP_{GB\text{-}P\_TYPE} = \lceil (4+4+22) \times 150 / (4096 \times 0.69) \rceil =$ **2**

Total cost = 257.271 + 165.800 + 2 = **381.623**

Exercise 12

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte len(P)=len(K)=4 byte NB = 101 u = 0.69

No projections on intermediate results

```
select N_NAME, count(*)
from TPCD.CUSTOMER, TPCD.NATION, TPCD.SUPPLIER
where C_NATIONKEY=S_NATIONKEY and C_NATIONKEY=N_NATIONKEY
GROUP BY N_NAME;
```

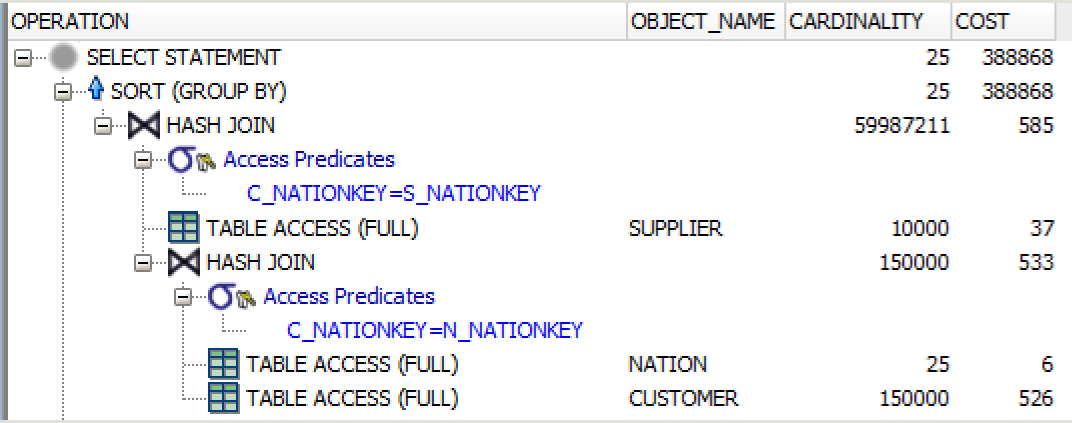
Exercise 12

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69

No projections on intermediate results

```
select n_name, count(*)
from TPCD.CUSTOMER, TPCD.NATION, TPCD.SUPPLIER
where C_NATIONKEY=S_NATIONKEY and C_NATIONKEY=N_NATIONKEY
GROUP BY N_NAME;
```



```
OPERATION                                  OBJECT_NAME  CARDINALITY    COST
SELECT STATEMENT                                                 25  388868
  SORT (GROUP BY)                                                25  388868
    HASH JOIN                                              59987211     585
      Access Predicates
        C_NATIONKEY=S_NATIONKEY
      TABLE ACCESS (FULL)                  SUPPLIER           10000      37
      HASH JOIN                                              150000     533
        Access Predicates
          C_NATIONKEY=N_NATIONKEY
        TABLE ACCESS (FULL)                NATION                25       6
        TABLE ACCESS (FULL)                CUSTOMER          150000     526
```

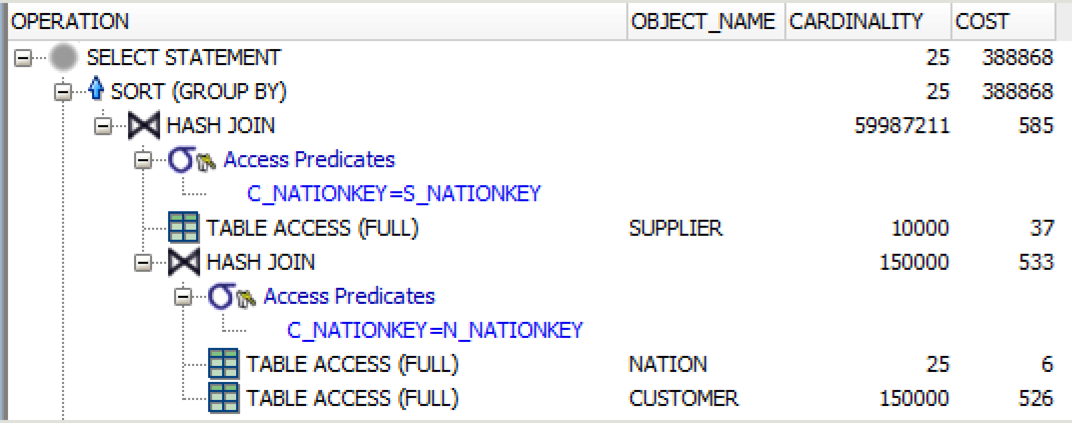
Exercise 12

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

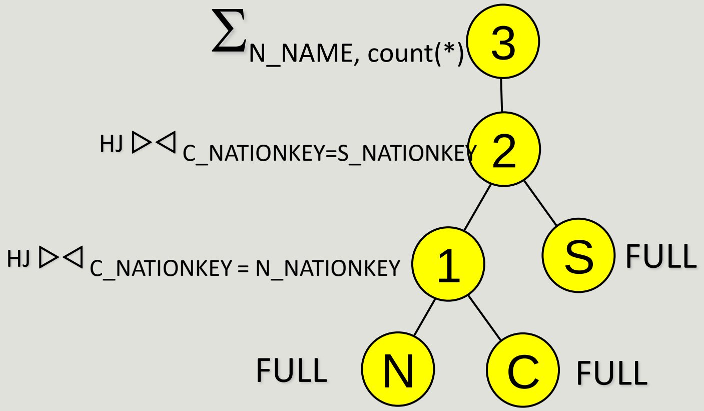
D = 4096 byte len(P)=len(K)=4 byte NB = 101 u = 0.69

No projections on intermediate results

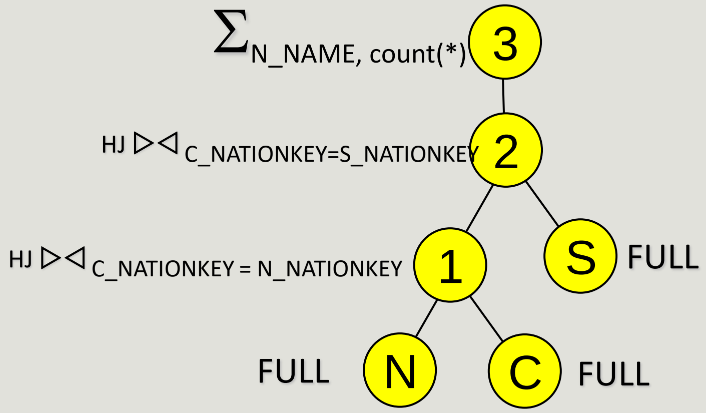
```
select n_name, count(*)
from TPCD.CUSTOMER, TPCD.NATION, TPCD.SUPPLIER
where C_NATIONKEY=S_NATIONKEY and C_NATIONKEY=N_NATIONKEY
GROUP BY N_NAME;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		25	388868
SORT (GROUP BY)		25	388868
HASH JOIN		59987211	585
Access Predicates			
C_NATIONKEY=S_NATIONKEY			
TABLE ACCESS (FULL)	SUPPLIER	10000	37
HASH JOIN		150000	533
Access Predicates			
C_NATIONKEY=N_NATIONKEY			
TABLE ACCESS (FULL)	NATION	25	6
TABLE ACCESS (FULL)	CUSTOMER	150000	526



$\Sigma_{\text{N\_NAME, count(*)}}$
3
HJ ▷◁ C_NATIONKEY=S_NATIONKEY
2
HJ ▷◁ C_NATIONKEY = N_NATIONKEY
1
S
FULL
FULL
N
C
FULL


$\Sigma_{\text{N\_NAME, count(*)}}$
3
HJ $\rhd\lhd_{\text{C\_NATIONKEY=S\_NATIONKEY}}$
2
HJ $\rhd\lhd_{\text{C\_NATIONKEY = N\_NATIONKEY}}$
1
S
FULL
FULL
N
C
FULL

Exercise 12

$NP_{NATION} = \lceil 25 \times 106/ (4096 \times 0{,}69) \rceil = 1$

$NP_{CUSTORMER} = \lceil 150.000 \times 159/ (4096 \times 0{,}69) \rceil = 8.439$

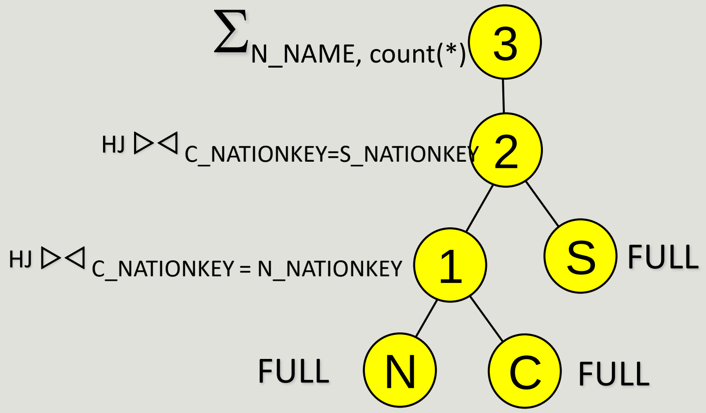
Cost $HJ_{C+N} = 8.439 + 1 =$ **8.440** (HJ since Nation fits the buffer)

$NP_{C+N} = \lceil 150.000 \times (159+106) / (4096 \times 0{,}69) \rceil = 14.065$

$NP_S = \lceil 10.000 \times 144/ (4096 \times 0{,}69) \rceil = 510$

Cost HHJ_{C+N+S} $3 \times (14.065 + 510)$ = **43.725**

Exercise 12


$\Sigma_{\text{N\_NAME, count(*)}}$
3
HJ $\rhd\lhd_{\text{C\_NATIONKEY=S\_NATIONKEY}}$
2
HJ $\rhd\lhd_{\text{C\_NATIONKEY = N\_NATIONKEY}}$
1
S
FULL
FULL
N
C
FULL

$NP_{C+N+S} = \lceil 60.000.000 \times (159+106+144) / (4096 \times 0{,}69) \rceil = 8.682.915$

The number of NP_{C+N+S} tuples is so high because the join condition is on the NATIONKEY field. The number of tuples must be calculated using the DB statistics and making assumptions of uniform distribution of probability:

- there are 25 countries, 10,000 suppliers and 150,000 customers
- so on average there will be 10.000 / 25 = 400 suppliers and 150.000 / 25 = 6.000 customers for each nation.
- so 6.000 x 400 x 25 = 60.000.000

You can verify the estimate, running the following query (results are very accurate since TPCD is a synthetic DB)

```
select count(*)
from TPCD.customer, TPCD.supplier
where s_nationkey=c_nationkey;
```

$\text{Sort GB}_{C+N+S} = 2 \times 8.682.915 \times (\lceil \log_{100} 8.682.915 \rceil + 1) = 2 \times 8.619.226 \times (4+1) = 86.192.260$

Total cost = 8.440 + 43.725 + 86.192.260 = 86.244.425

Exercise 13

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69   No projections on intermediate results

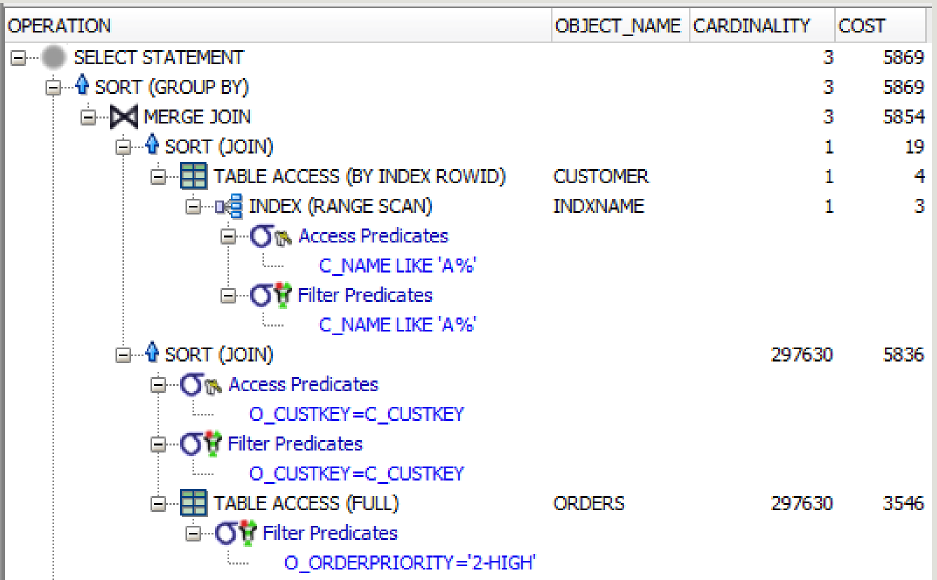
```
select /*+ USE_MERGE(ORDERS,CUSTOMER)*/ O_CLERK, sum(O_TOTALPRICE)
from TPCD.ORDERS,TPCD.CUSTOMER
where O_CUSTKEY=C_CUSTKEY AND C_NAME LIKE 'A%' AND O_ORDERPRIORITY='2-HIGH'
group by O_CLERK;
```

Exercise 13

After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

D = 4096 byte   len(P)=len(K)=4 byte   NB = 101   u = 0.69   No projections on intermediate results

```
select /*+ USE_MERGE(ORDERS,CUSTOMER)*/ O_CLERK, sum(O_TOTALPRICE)
from TPCD.ORDERS,TPCD.CUSTOMER
where O_CUSTKEY=C_CUSTKEY AND C_NAME LIKE 'A%' AND O_ORDERPRIORITY='2-HIGH'
group by O_CLERK;
```



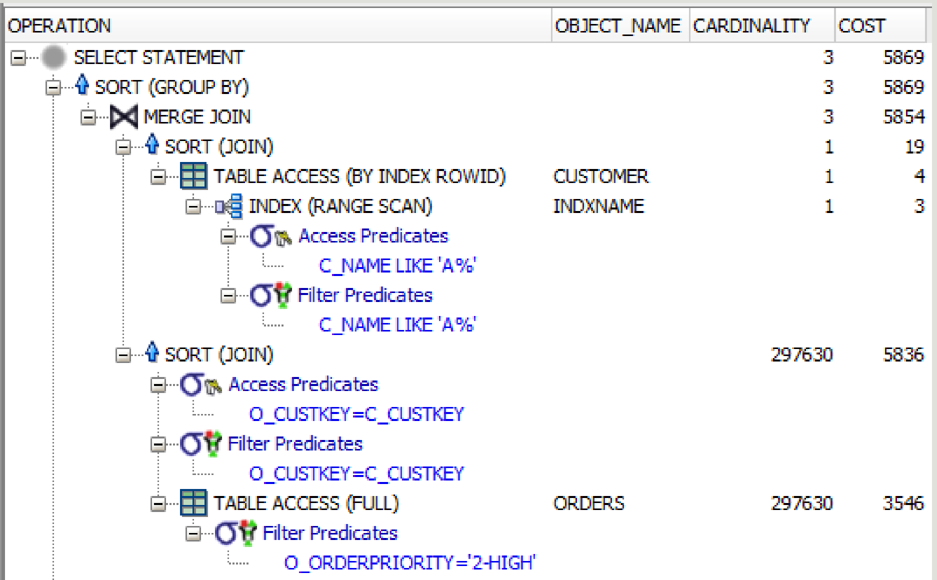
```
OPERATION                                  OBJECT_NAME  CARDINALITY   COST
SELECT STATEMENT                                                  3   5869
  SORT (GROUP BY)                                                 3   5869
    MERGE JOIN                                                    3   5854
      SORT (JOIN)                                                 1     19
        TABLE ACCESS (BY INDEX ROWID)      CUSTOMER               1      4
          INDEX (RANGE SCAN)               INDXNAME               1      3
            Access Predicates
              C_NAME LIKE 'A%'
            Filter Predicates
              C_NAME LIKE 'A%'
      SORT (JOIN)                                            297630   5836
        Access Predicates
          O_CUSTKEY=C_CUSTKEY
        Filter Predicates
          O_CUSTKEY=C_CUSTKEY
        TABLE ACCESS (FULL)                ORDERS            297630   3546
          Filter Predicates
            O_ORDERPRIORITY='2-HIGH'
```

Exercise 13

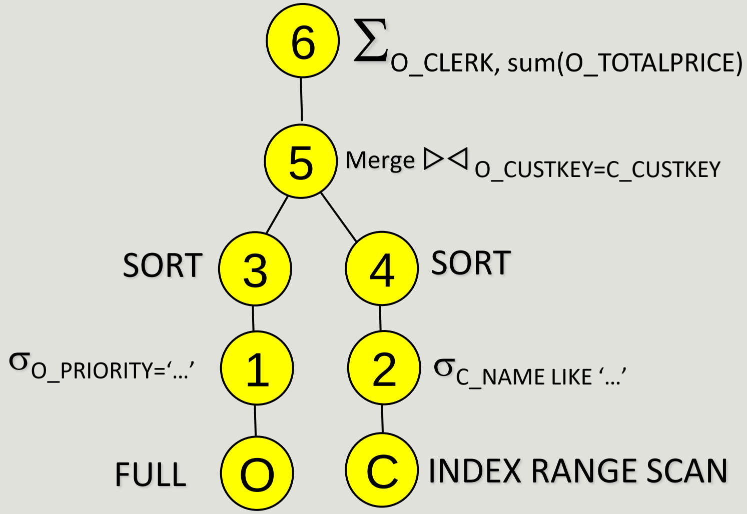
After drawing the execution tree of the optimizer for the query, compute the execution cost assuming that:

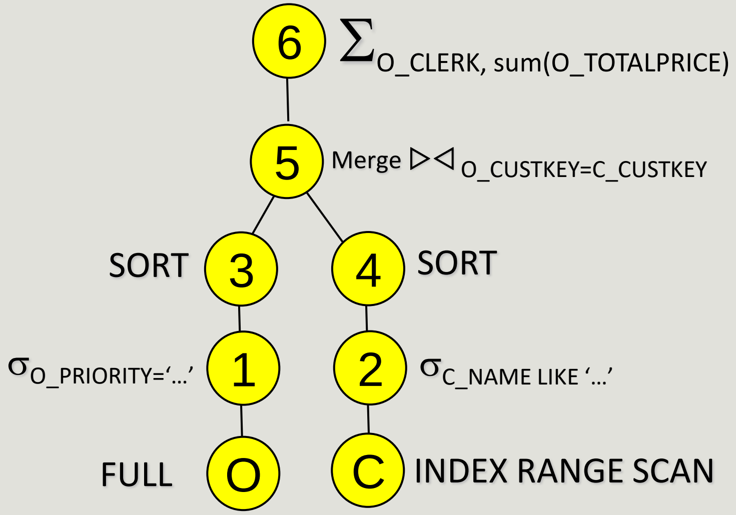
D = 4096 byte len(P)=len(K)=4 byte NB = 101 u = 0.69 No projections on intermediate results

```
select /*+ USE_MERGE(ORDERS,CUSTOMER)*/ O_CLERK, sum(O_TOTALPRICE)
from TPCD.ORDERS,TPCD.CUSTOMER
where O_CUSTKEY=C_CUSTKEY AND C_NAME LIKE 'A%' AND O_ORDERPRIORITY='2-HIGH'
group by O_CLERK;
```



OPERATION	OBJECT_NAME	CARDINALITY	COST
SELECT STATEMENT		3	5869
SORT (GROUP BY)		3	5869
MERGE JOIN		3	5854
SORT (JOIN)		1	19
TABLE ACCESS (BY INDEX ROWID)	CUSTOMER	1	4
INDEX (RANGE SCAN)	INDXNAME	1	3
Access Predicates			
C_NAME LIKE 'A%'			
Filter Predicates			
C_NAME LIKE 'A%'			
SORT (JOIN)		297630	5836
Access Predicates			
O_CUSTKEY=C_CUSTKEY			
Filter Predicates			
O_CUSTKEY=C_CUSTKEY			
TABLE ACCESS (FULL)	ORDERS	297630	3546
Filter Predicates			
O_ORDERPRIORITY='2-HIGH'			


6
$\Sigma_{\text{O\_CLERK, sum(O\_TOTALPRICE)}}$
5
Merge ▷◁ O_CUSTKEY=C_CUSTKEY
SORT
3
4
SORT
σO_PRIORITY='…'
1
2
σC_NAME LIKE '…'
FULL
O
C
INDEX RANGE SCAN



6 $\Sigma_{O\_CLERK,\ sum(O\_TOTALPRICE)}$

5 Merge $\triangleright\triangleleft_{O\_CUSTKEY=C\_CUSTKEY}$

SORT 3

4 SORT

$\sigma_{O\_PRIORITY='...'}$ 1

2 $\sigma_{C\_NAME\ LIKE\ '...'}$

FULL O

C INDEX RANGE SCAN

Exercise 13

$NP_{CUSTORMER} = \lceil 150.000 \times 159/ (4096 \times 0,69) \rceil = 8.439$

$NP_{ORDERS} = \lceil 1.500.000 \times 106/ (4096 \times 0,69) \rceil = 56.259$

Sel(C_NAME LIKE 'A%')= 1/26

Sel(O_ORDERPRIORITY='2-HIGH')=1/5

$ET_{Sel(C\_NAME\ LIKE\ '...')} = \lceil 150.000 \times 1/26 \rceil = 5.770$

$NP_{O\text{-}FILTERED} = \lceil 1.500.000/5 \times 106/ (4096 \times 0,69) \rceil = 11.252$

$NP_{C\text{-}FILTERED} = \lceil 150.000 \times 1/26 \times 159/ (4096 \times 0,69) \rceil = 325$

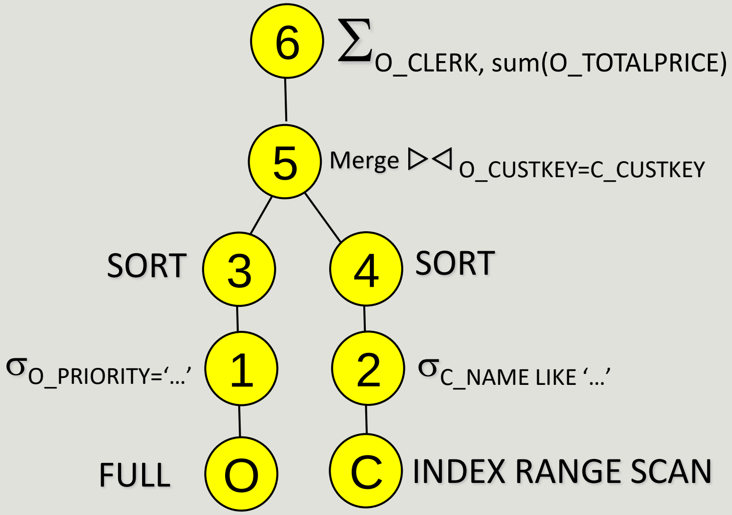
SORT(ORDERS)= $56.259 + 11.252 + 2 \times 11.252 \times \lceil \log_{100} 11.252 \rceil = 56.259 + (1 + 2 \times 3) \times 11.252 =$ **135.023**

$NL_{C\_NAME} = \lceil (4 \times 150.000 + 4 \times 150.000) / (4096 \times 0,69) \rceil = 425$

Uncl. access to C = $2 + \lceil 425 / 26 \rceil + 5.770 \times \Phi(1, 8.439) = 2 + 17 + 5.770 = 5.788$

SORT(CUSTOMER)= $5.788 + 325 + 2 \times 325 \times \lceil \log_{100} 325 \rceil = 5.788 + 325 + 325 \times 4 =$ **7.413**

Cost SORT-MERGE JOIN = (135.023 + 7.413) + (325 + 11.252) = **154.013**


6
$\Sigma_{O\_CLERK,\ sum(O\_TOTALPRICE)}$
5
Merge $\triangleright\triangleleft_{O\_CUSTKEY=C\_CUSTKEY}$
SORT
3
4
SORT
$\sigma_{O\_PRIORITY='…'}$
1
2
$\sigma_{C\_NAME\ LIKE\ '…'}$
FULL
O
C
INDEX RANGE SCAN

Exercise 13

$ET_{C\text{-}O} = \lceil 1.500.000 \times 1/26 \times 1/\ 5 \rceil = 11.539$

$NP_{C\text{-}O} = \lceil 11.539 \times (106+159)/\ (4096 \times 0{,}69) \rceil = 1.082$

$GB_{O\_CLERK} = 2 \times 1.082 \times (\lceil \log_{100} 1.082 \rceil + 1) = 1.082 \times 6 = \mathbf{6.492}$

Total cost = **153.688 + 6.492 = 160.180**