Συστήματα Διαχείρισης και Ανάλυσης Δεδομένων

1ο Project 2023

Βεάζογλου Γεώργιος 3190347

Ζήτημα Πρώτο:

Ευρετήριο idx_orders:

```
create index idx_orders on orders(orderdate, custkey, orderkey)
```

```
create index idx_lineitem on lineitem(shipdate) include(price)
```

Loaded σελίδες:

	Χωρίς ευρετήριο	Με ευρετήριο
Logical reads	79.725	4.123
Physical reads	5	4
Read-ahead reads	78.575	3.932

Χωρίς ευρετήριο:

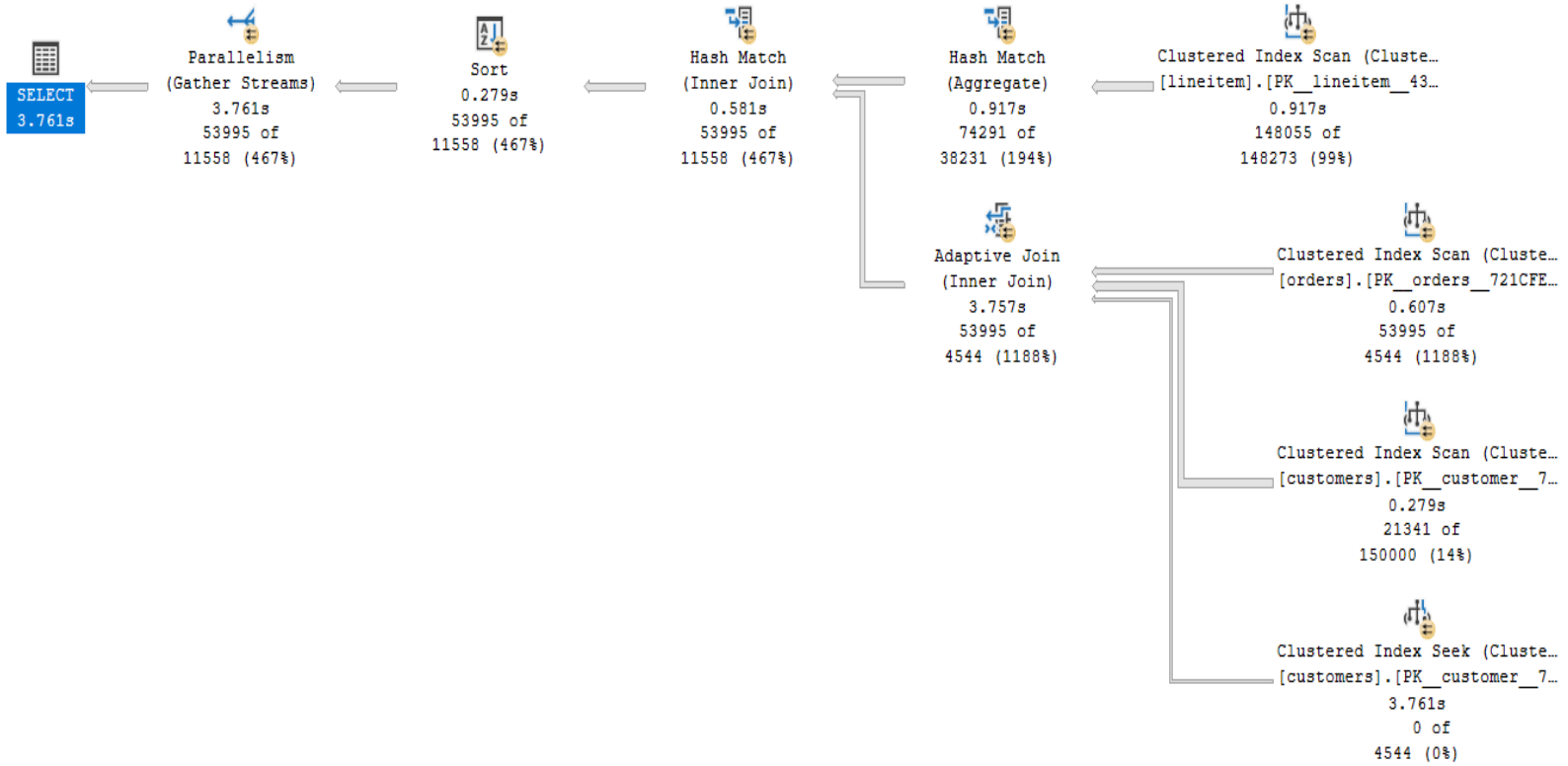
```
(53995 rows affected)
Table 'customers'. Scan count 5, logical reads 2685, physical reads 3, page server reads 0, read-ahead reads 2554
Table 'orders'. Scan count 5, logical reads 16961, physical reads 1, page server reads 0, read-ahead reads 16717
Table 'lineitem'. Scan count 5, logical reads 59779, physical reads 1, page server reads 0, read-ahead reads 59304
```

Με ευρετήριο:

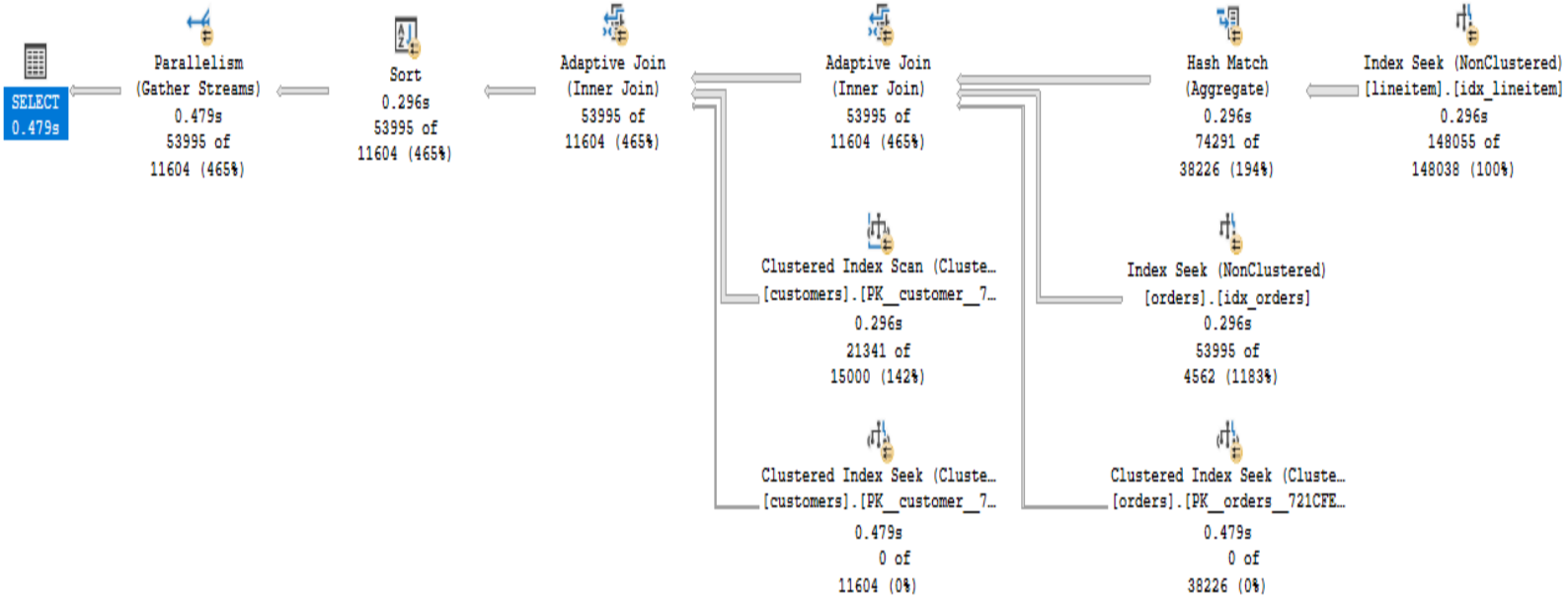
```
(53995 rows affected)
Table 'customers'. Scan count 5, logical reads 2685, physical reads 2, page server reads 0, read-ahead reads 2562,
Table 'orders'. Scan count 5, logical reads 1010, physical reads 1, page server reads 0, read-ahead reads 967, pa
Table 'lineitem'. Scan count 5, logical reads 428, physical reads 1, page server reads 0, read-ahead reads 403, p
```

Πλάνα εκτέλεσης:

Χωρίς ευρετήριο:



Με ευρετήριο:



Ζήτημα Δεύτερο:

Ευρετήριο idx_partsupp:

```
create index idx_partsupp on partsupp(partkey, suppkey)
```

Ευρετήριο idx_parts:

```
create index idx_parts on parts(ptype, psize)
```

Loaded σελίδες:

	Χωρίς ευρετήριο	Με ευρετήριο
Logical reads	3.722	760
Physical reads	53	57
Read-ahead reads	3.010	176

Χωρίς ευρετήριο:

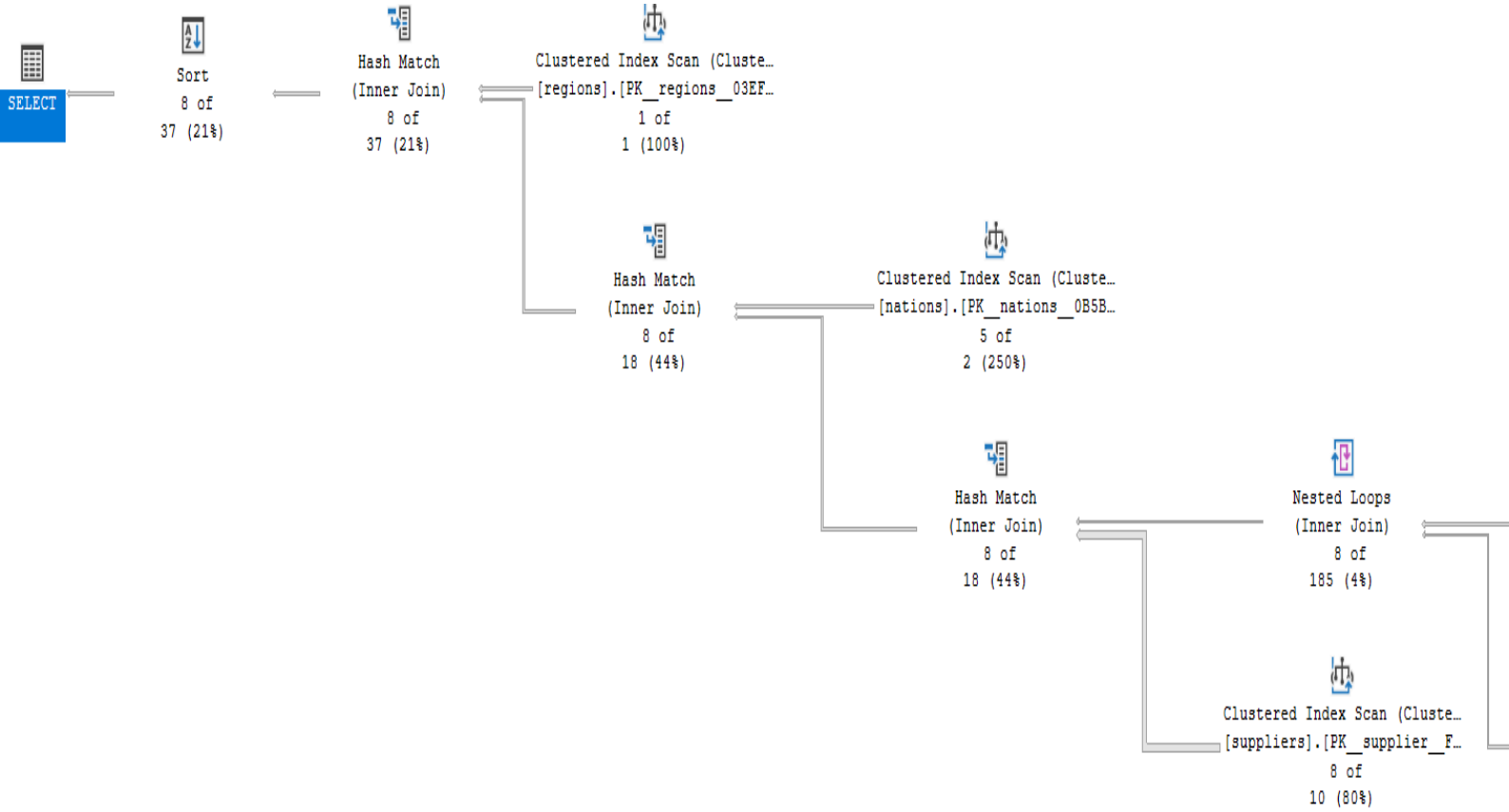
```
(8 rows affected)
Table 'suppliers'. Scan count 1, logical reads 479, physical reads 1, page server reads 0, read-ahead reads 219
Table 'nations'. Scan count 1, logical reads 170, physical reads 1, page server reads 0, read-ahead reads 0
Table 'regions'. Scan count 1, logical reads 170, physical reads 1, page server reads 0, read-ahead reads 0
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0
Table 'partsupp'. Scan count 29, logical reads 108, physical reads 47, page server reads 0, read-ahead reads 0
Table 'parts'. Scan count 1, logical reads 2795, physical reads 3, page server reads 0, read-ahead reads 2791
```

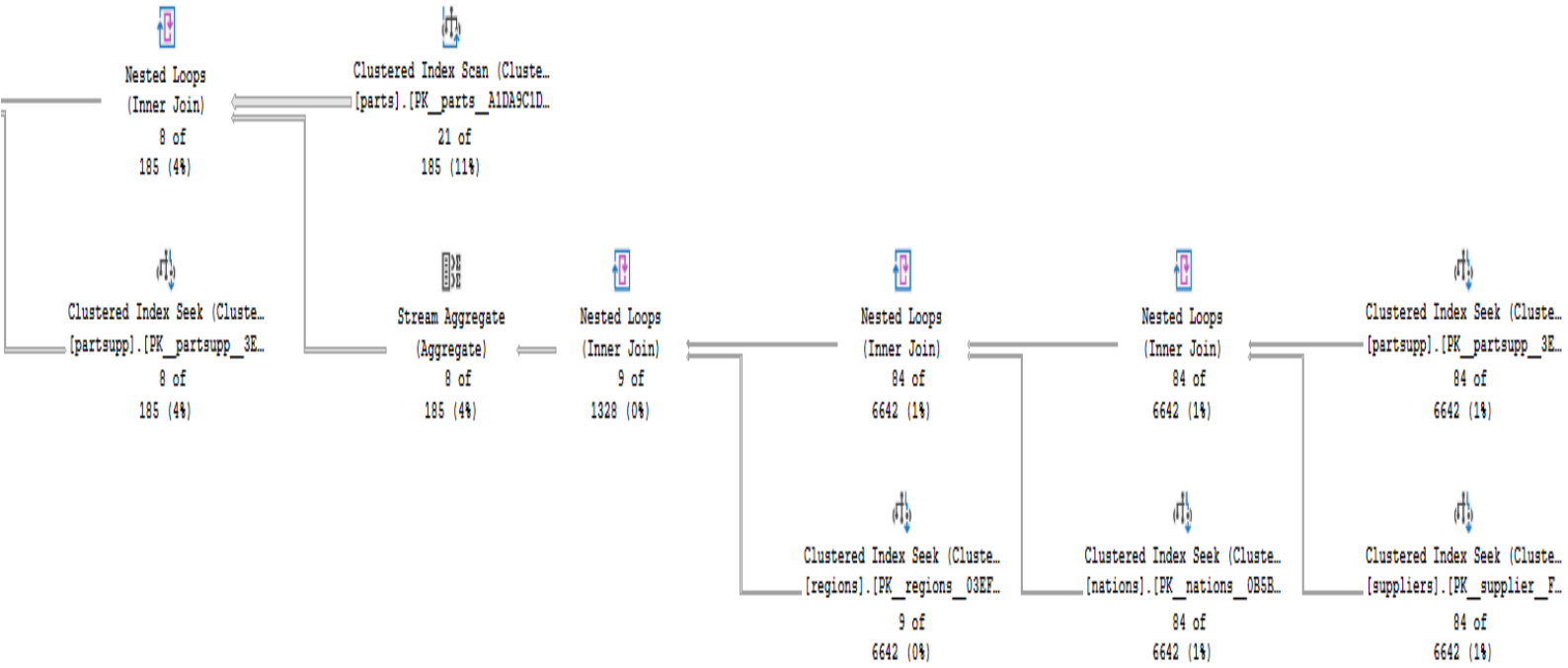
Με ευρετήριο:

```
(8 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0
Table 'regions'. Scan count 0, logical reads 184, physical reads 1, page server reads 0, read-ahead reads 0
Table 'nations'. Scan count 0, logical reads 184, physical reads 1, page server reads 0, read-ahead reads 0
Table 'suppliers'. Scan count 0, logical reads 184, physical reads 22, page server reads 0, read-ahead reads 0
Table 'partsupp'. Scan count 29, logical reads 93, physical reads 29, page server reads 0, read-ahead reads 0
Table 'parts'. Scan count 1, logical reads 115, physical reads 4, page server reads 0, read-ahead reads 176
```

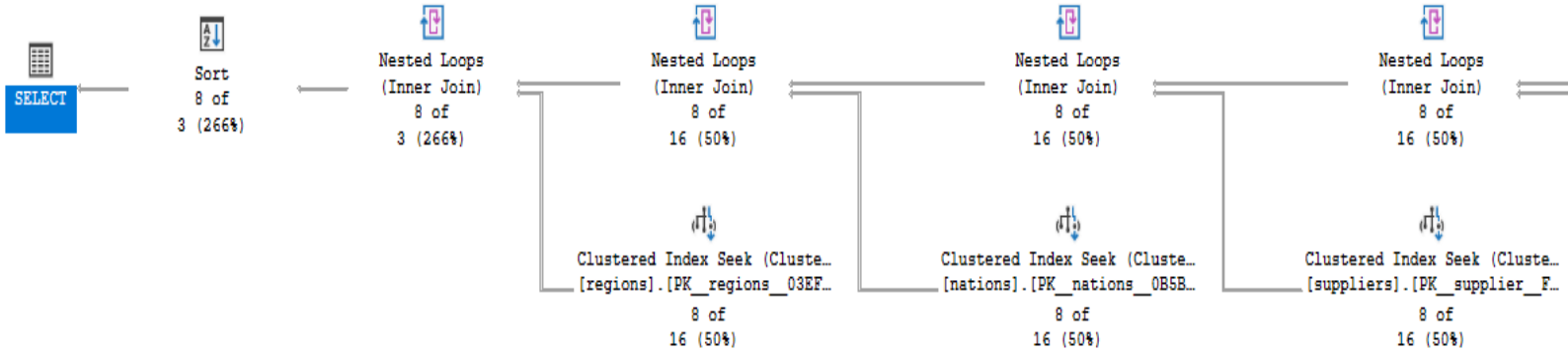
Πλάνα εκτέλεσης:

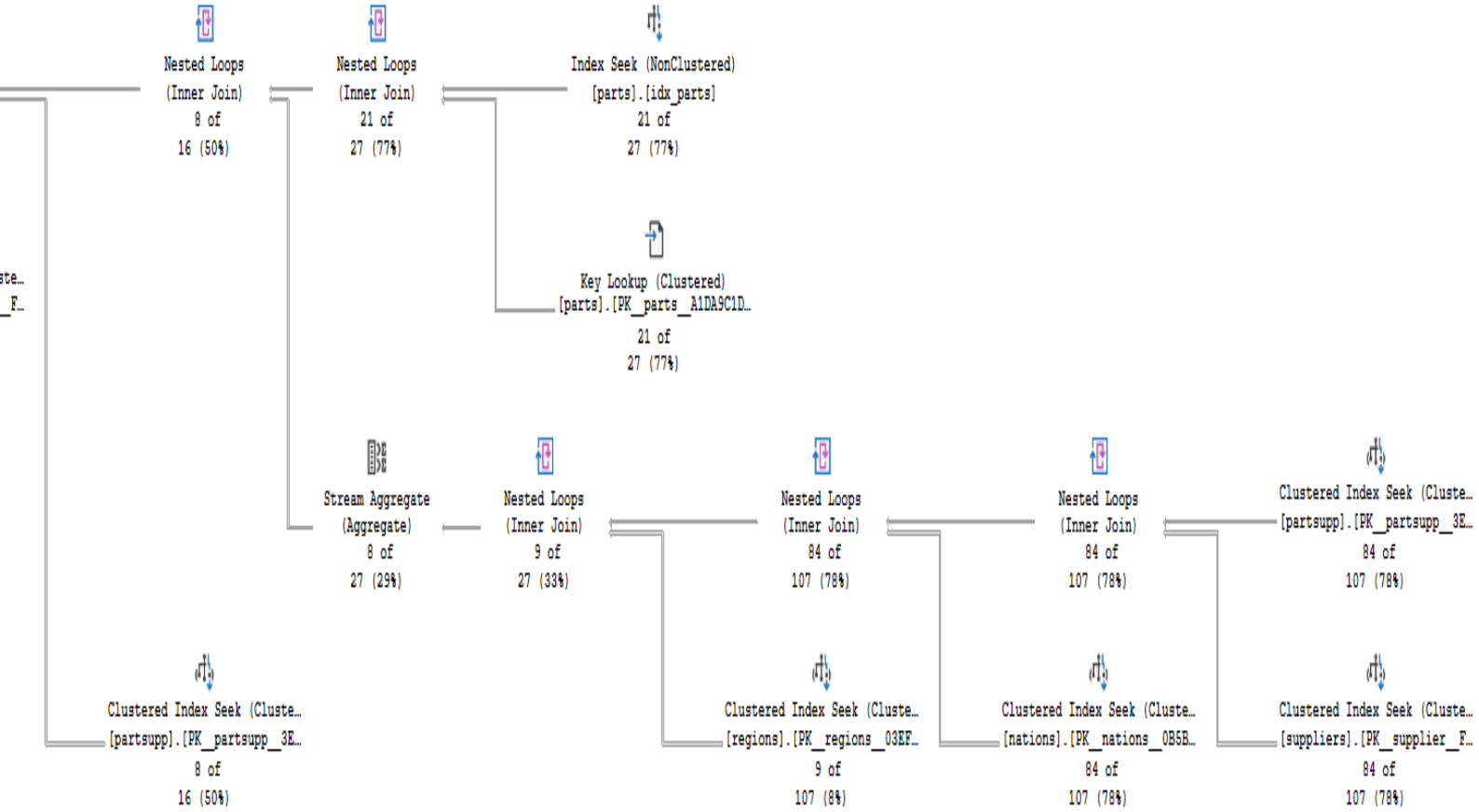
Χωρίς ευρετήριο:





Με ευρετήριο:





Ζήτημα Τρίτο:

Στο επερώτημα του προγραμματιστή γίνεται σύζευξη μεταξύ των εγγραφών των δύο πινάκων και στη συνέχεια επιλέγονται οι επιθυμητές βάσει έτους.

Αντί να γίνει σύζευξη όλων των εγγραφών των πινάκων, μπορούμε να προσεγγίσουμε το πρόβλημα αντίστροφα. Δηλαδή, μπορούμε πρώτα να φιλτράρουμε τις εγγραφές του πίνακα orders, να χρησιμοποιήσουμε μόνο εκείνες που ικανοποιούν τον περιορισμό (συγκεκριμένο έτος) και τέλος, να γίνει η σύζευξη μεταξύ των δύο πινάκων.

Έτσι προκύπτει το νέο query:

```
select market_segment, sum(totalprice)
from customers
join (
       select custkey, totalprice from orders
       where year(orderdate) = 1996) as ord
       on customers.custkey = ord.custkey
group by customers.market_segment
```

Στη συνέχεια, δημιουργούμε δύο indexes:

```
create index idx_customers on customers(market_segment, custkey)
```

```
create index idx_orders on orders(orderdate, custkey, totalprice)
```

Loaded σελίδες:

	Χωρίς ευρετήριο	Με ευρετήριο
Logical reads	19.646	5.874
Physical reads	3	4
Read-ahead reads	19.290	4.536

Χωρίς ευρετήριο:

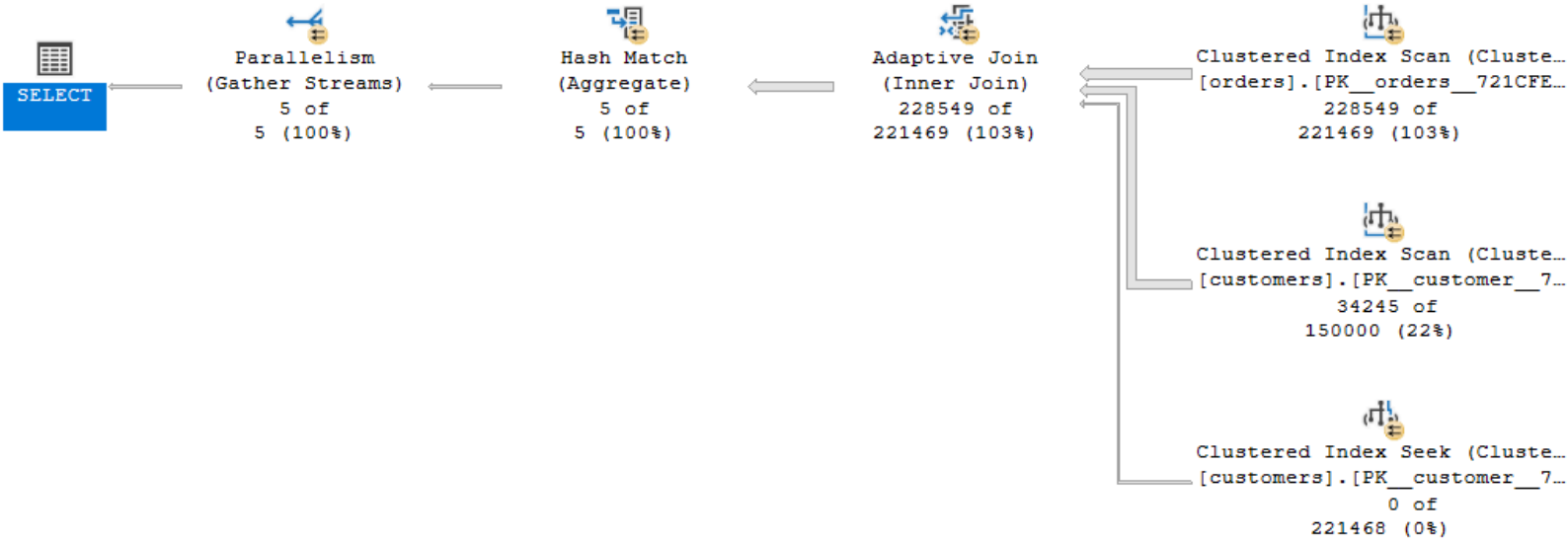
```
(5 rows affected)
Table 'customers'. Scan count 5, logical reads 2685, physical reads 2, page server reads 0, read-ahead reads 2562
Table 'orders'. Scan count 5, logical reads 16961, physical reads 1, page server reads 0, read-ahead reads 16728
```

Με ευρετήριο:

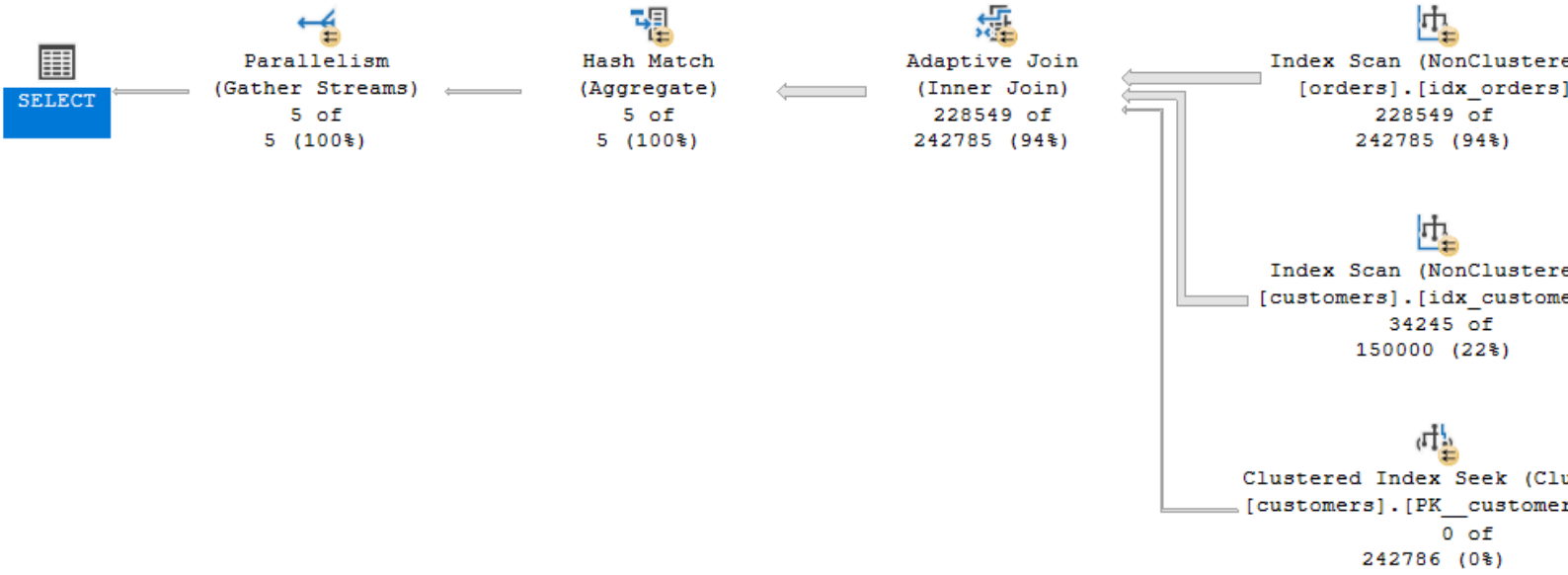
```
(5 rows affected)
Table 'customers'. Scan count 5, logical reads 1716, physical reads 3, page server reads 0, read-ahead reads 428
Table 'orders'. Scan count 5, logical reads 4158, physical reads 1, page server reads 0, read-ahead reads 4108
```

Πλάνα εκτέλεσης:

Χωρίς ευρετήριο:



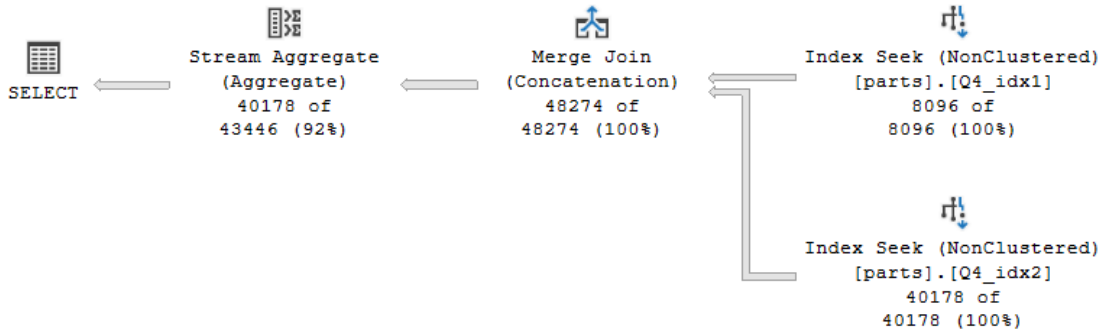
Με ευρετήριο:



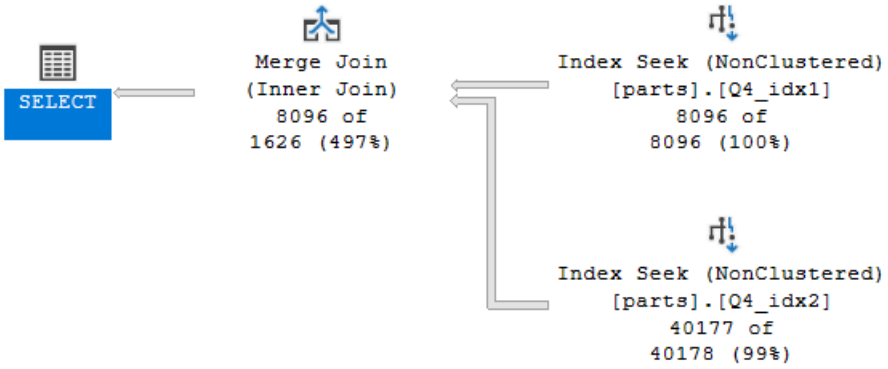
Ζήτημα Τέταρτο:

1.

Πλάνο εκτέλεσης query a:



Πλάνο εκτέλεσης query b:



Παρατηρούμε ότι και στις δύο περιπτώσεις πριν το operation Merge Join, πραγματοποιούνται δύο operations Index Seek τα οποία χρησιμοποιούν τα non clustered ευρετήρια που δημιουργήθηκαν από τον προγραμματιστή με ονόματα Q4_idx1 και Q4_idx2. Άρα τα παραπάνω ευρετήρια χρησιμοποιούνται στην εκτέλεση των επερωτημάτων του προγραμματιστή.

2.

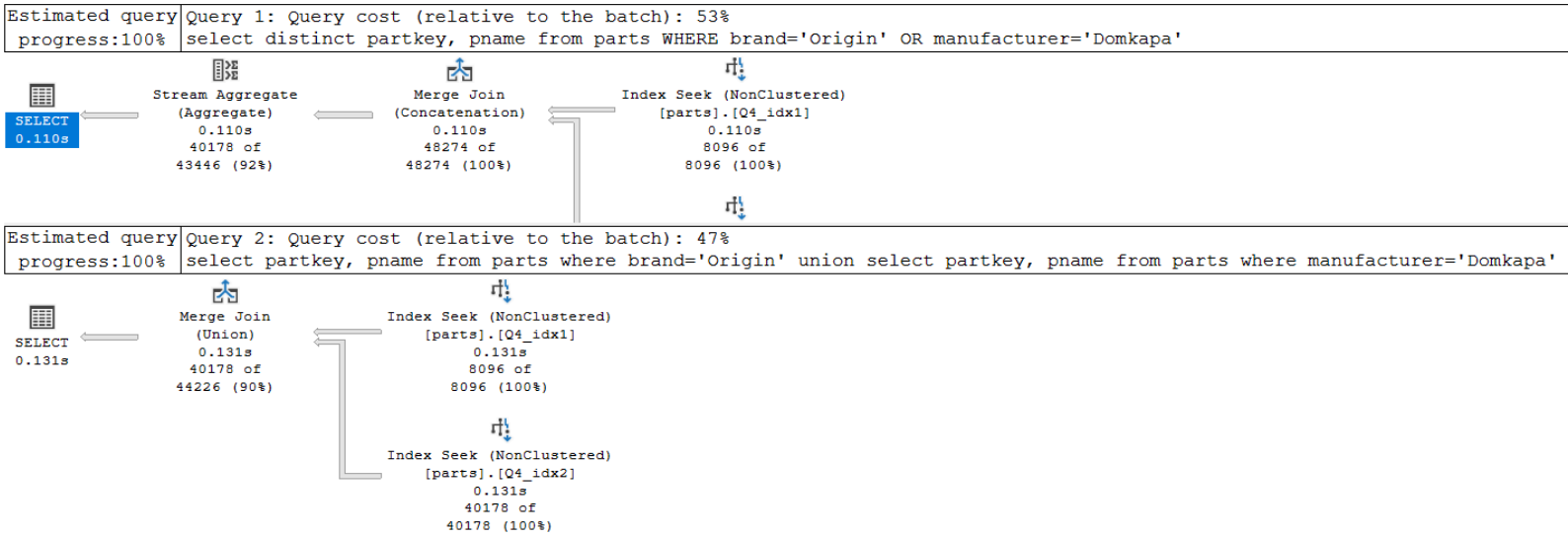
Query a.

Εάν τροποποιήσουμε το επερώτημα b και αντί για την τομή (intersect) πάρουμε την ένωση (union), θα καταλήξουμε σε ένα νέο επερώτημα που θα μας δώσει το ίδιο αποτέλεσμα με το πρώτο. Έτσι προκύπτει το query a':

```
select partkey, pname
 from parts
where brand='Origin'
union
select partkey, pname
 from parts
```

```
where manufacturer='Domkapa'
```

Τα πλάνα εκτέλεσης των δύο επερωτημάτων είναι τα εξής:



Από τα πλάνα εκτέλεσης παρατηρούμε ότι το query 2, δηλ. το a', κοστίζει λιγότερο.

Query b.

Τροποποιούμε το ερώτημα ως εξής:

```
select partkey, pname
from parts
where brand = 'Origin' and manufacturer = 'Domkapa'
```

Και δημιουργούμε το κατάλληλο index:

```
create index Q4_idx3 on parts(brand, manufacturer) include (pname)
```

Ο συνδυασμός των δύο οδηγεί στην εξής βελτιστοποίηση:

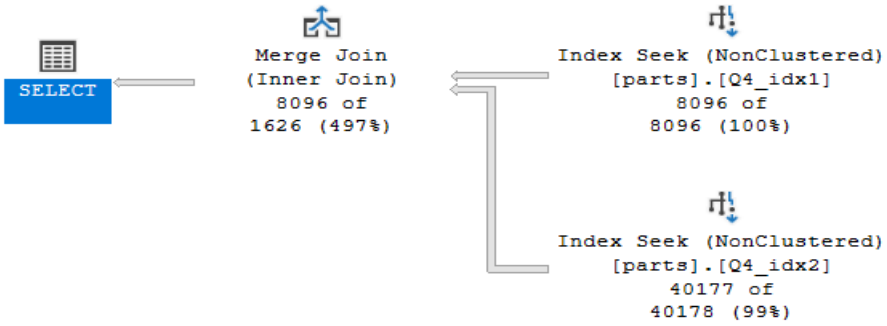
Reads αρχικής εκδοχής:

```
(8096 rows affected)
Table 'parts'. Scan count 2, logical reads 345, physical reads 5, page server reads 0, read-ahead reads 358
```

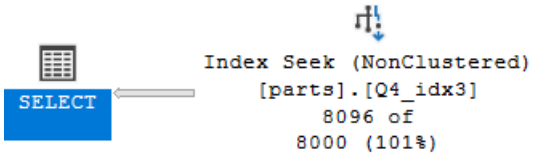
Reads βελτιωμένης εκδοχής:

```
(8096 rows affected)
Table 'parts'. Scan count 1, logical reads 68, physical reads 2, page server reads 0, read-ahead reads 65
```

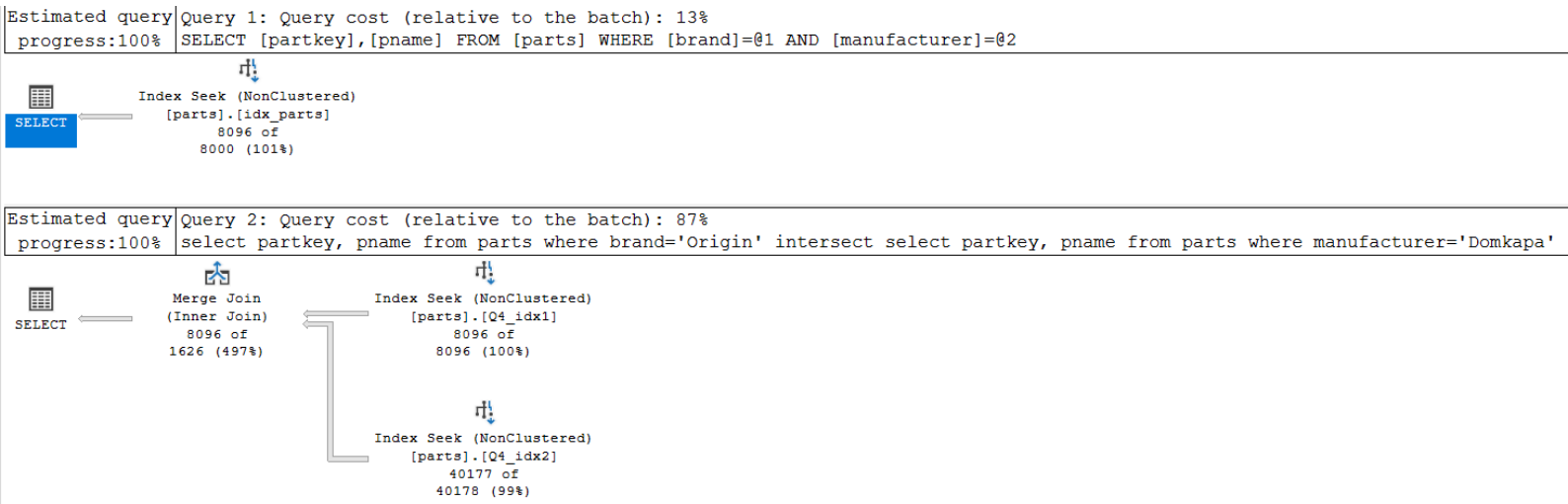
Πλάνο εκτέλεσης αρχικής εκδοχής:



Πλάνο εκτέλεσης βελτιωμένης εκδοχής:



Σύγκριση βελτιωμένου (query 1) και αρχικού (query 2) μέσω του πλάνου εκτέλεσης:



Ζήτημα Πέμπτο:

1.

Εμφάνισε το σύνολο των πελατών και τη συνολική αξία των παραγγελιών ανά κράτος. Τύπωσε τα αποτελέσματα σε αλφαβητική σειρά.

```
select nation, count(distinct(customers.custkey)) as
total_customers,sum(totalprice) as total_price
from nations, orders, customers
where customers.custkey = orders.custkey
and customers.nationkey = nations.nationkey
group by nation
order by nation asc
```

Ευρετήρια που δημιουργήθηκαν:

```
create index idx_customers on customers(nationkey, custkey)
create index idx_orders on orders(orderkey, custkey, totalprice)
```

Loaded σελίδες:

	Χωρίς ευρετήριο	Με ευρετήριο
Logical reads	19.650	4.371
Physical reads	4	3
Read-ahead reads	19.289	3.808

Χωρίς ευρετήριο:

```
(25 rows affected)
Table 'orders'. Scan count 5, logical reads 16961, physical reads 1, page server reads 0, read-ahead reads 16727,
Table 'customers'. Scan count 5, logical reads 2685, physical reads 2, page server reads 0, read-ahead reads 2562
Table 'nations'. Scan count 5, logical reads 4, physical reads 1, page server reads 0, read-ahead reads 0, page s
```

Με ευρετήριο:

```
(25 rows affected)
Table 'orders'. Scan count 5, logical reads 3583, physical reads 1, page server reads 0, read-ahead reads 3547,
Table 'customers'. Scan count 5, logical reads 784, physical reads 1, page server reads 0, read-ahead reads 261
Table 'nations'. Scan count 5, logical reads 4, physical reads 1, page server reads 0, read-ahead reads 0, page
```

2.

Ποιος είναι ο συνολικός αριθμός προϊόντων που έχουν αγοραστεί ανά τύπο προϊόντος. Εμφάνισε τα αποτελέσματα σε αύξουσα σειρά.

```
select ptype, count(*) as total_products
from lineitem
join parts on lineitem.partkey = parts.partkey
group by ptype
order by total_products asc
```

Ευρετήρια που δημιουργήθηκαν:

```
create index idx_lineitem on lineitem(partkey)
create index idx_parts on parts(partkey) include (ptype)
```

Loaded σελίδες:

	Χωρίς ευρετήριο	Με ευρετήριο
Logical reads	62.758	10.750
Physical reads	3	4
Read-ahead reads	62.093	868

Χωρίς ευρετήριο:

```
(150 rows affected)
Table 'parts'. Scan count 5, logical reads 2935, physical reads 2, page server reads 0, read-ahead reads 2798, pag
Table 'lineitem'. Scan count 5, logical reads 59823, physical reads 1, page server reads 0, read-ahead reads 59295
```

Με ευρετήριο:

```
(150 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, read-ahead reads 0,
Table 'lineitem'. Scan count 1, logical reads 9885, physical reads 1, page server reads 0, read-ahead reads
Table 'parts'. Scan count 1, logical reads 865, physical reads 3, page server reads 0, read-ahead reads 868,
```