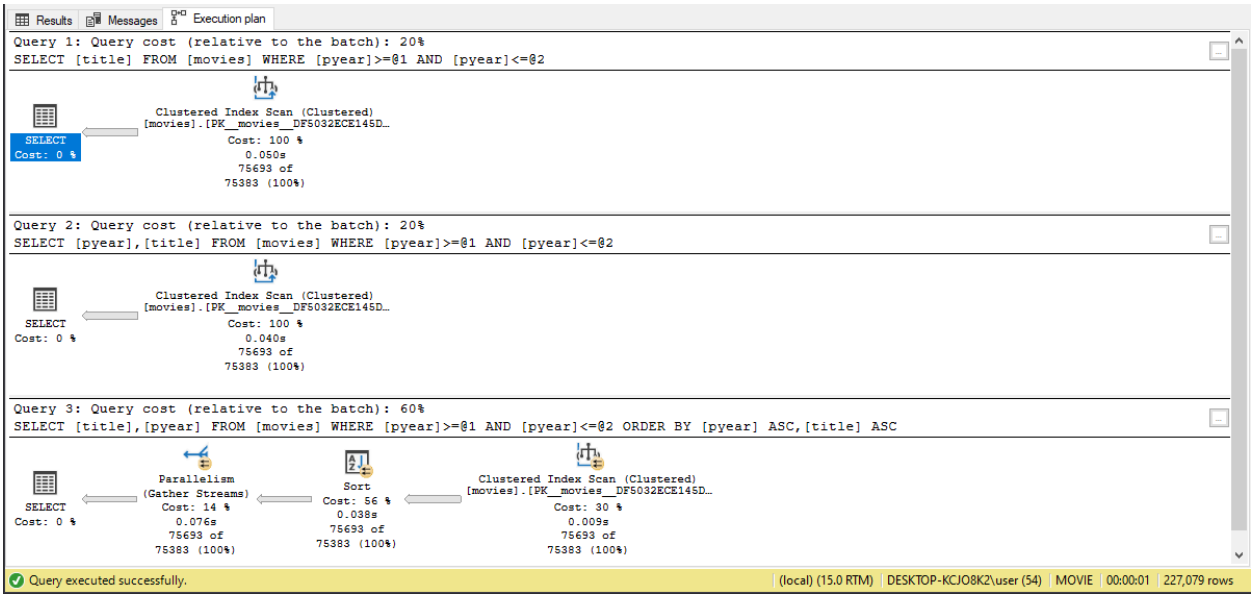
Project A

Δημήτριος Τσιομπίκας AM : 3180223

Ζήτημα Πρώτο

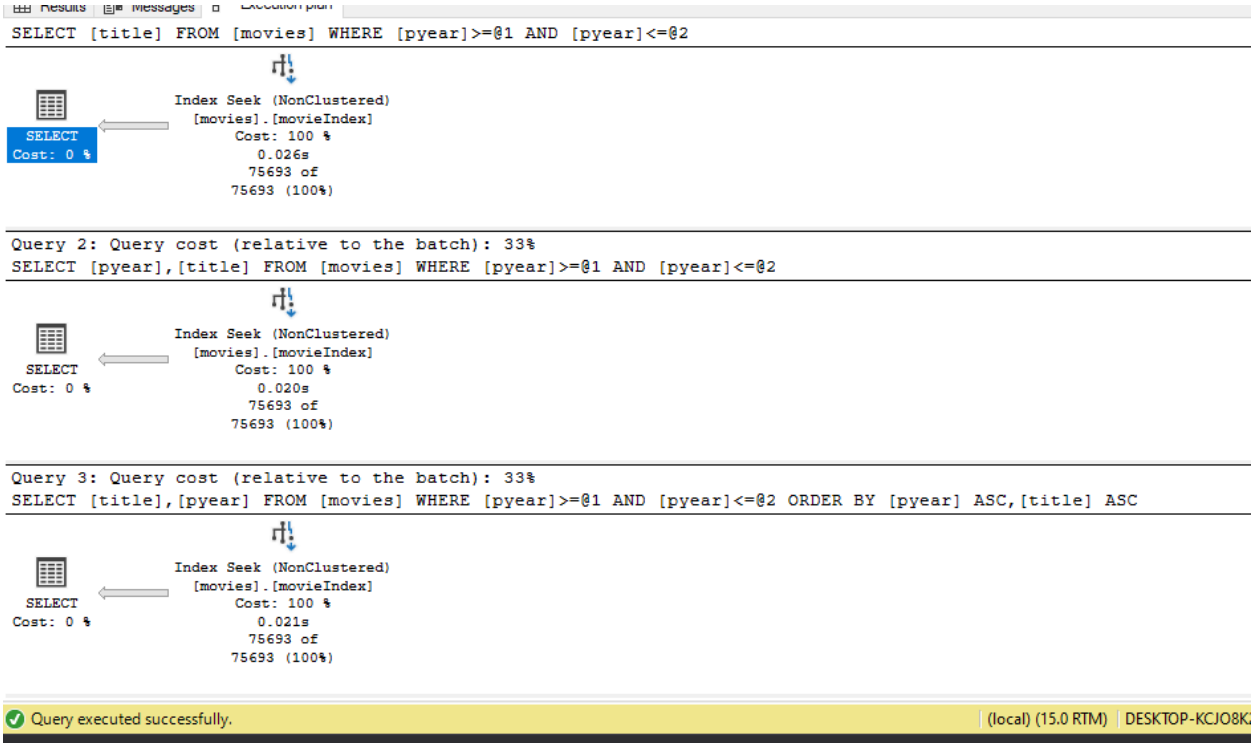
Άσκηση 1

Αποτελέσματα χωρίς ευρετήριο :




```
(75693 rows affected)
Table 'movies'. Scan count 1, logical reads 1918, physical reads 2, p

(75693 rows affected)
Table 'movies'. Scan count 1, logical reads 1918, physical reads 0, p

(75693 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, p
Table 'movies'. Scan count 5, logical reads 2012, physical reads 0, p
```

Αποτελέσματα με ευρετήριο :




```
(75693 rows affected)
Table 'movies'. Scan count 1, logical reads 351, physical reads 2,

(75693 rows affected)
Table 'movies'. Scan count 1, logical reads 351, physical reads 0,

(75693 rows affected)
Table 'movies'. Scan count 1, logical reads 351, physical reads 0,
```

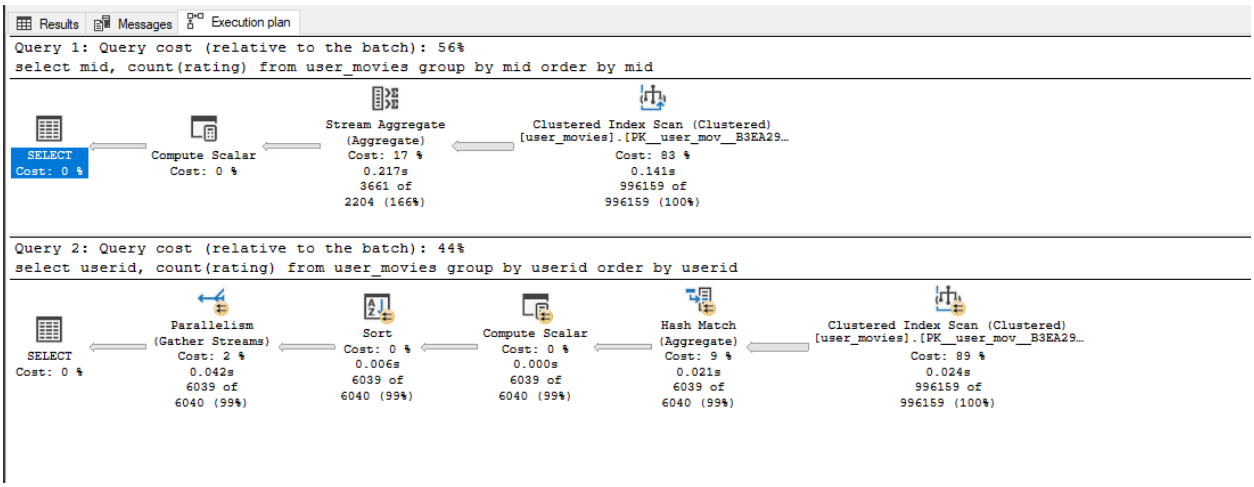
Παρατηρούμε βελτίωση στο χρόνο των επερωτημάτων και ειδικά στο 3ο επερώτημα. Οι εντολές που χρησιμοποιήθηκαν ήταν οι εξής :

```
/* Ζήτημα Πρώτο */

checkpoint
dbcc dropcleanbuffers

SET statistics IO ON;

CREATE INDEX movieIndex ON movies(pyear,title);

DROP INDEX movieIndex ON movies;


BEGIN
select title from movies where pyear between 1990 and 2000

select pyear, title from movies where pyear between 1990 and 2000

select title, pyear from movies where pyear between 1990 and 2000
order by pyear, title
END
```

Δοκίμασα τις εναλλαγές pyear , title και title,pyear στο ευρετήριο και αποδείχτηκε καλύτερη η pyear,title , εξασφαλίζοντας τον λιγότερο χρόνο καθώς χρησιμοποιεί την στήλη pyear από το where και το title για το select.

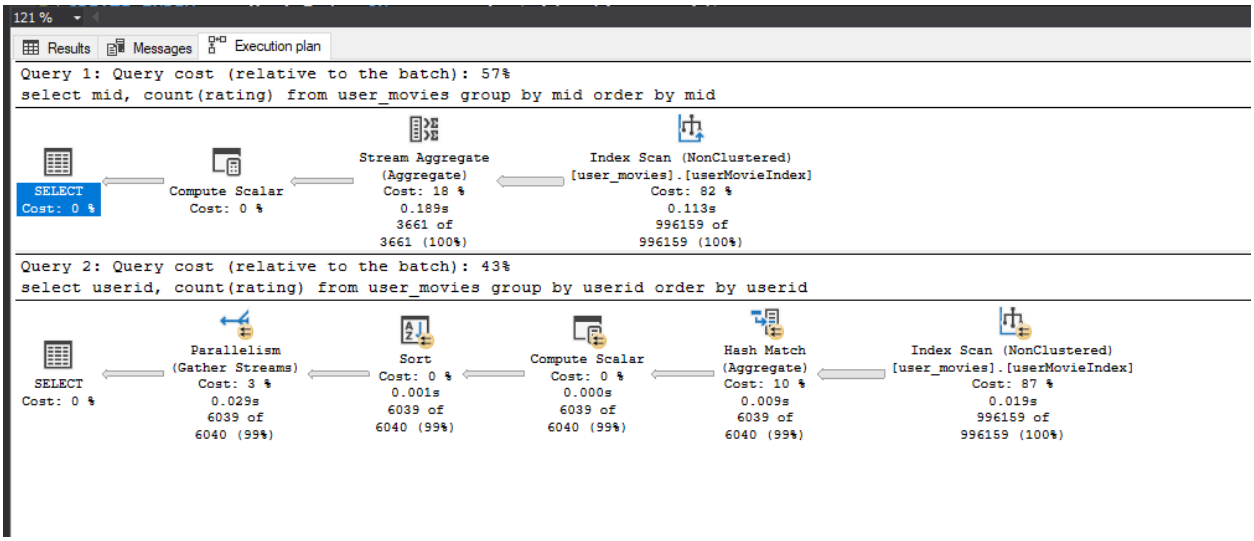
Άσκηση 2

Αποτελέσματα χωρίς ευρετήριο :




```
(3661 rows affected)
Table 'user_movies'. Scan count 1, logical reads 2601, physical reads 0, page

(6039 rows affected)
Table 'user_movies'. Scan count 5, logical reads 2733, physical reads 0, page
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page serv
```

Αποτελέσματα με ευρετήριο:



```
(3661 rows affected)
Table 'user_movies'. Scan count 1, logical reads 2231, physical r

(6039 rows affected)
Table 'user_movies'. Scan count 5, logical reads 2342, physical r
Table 'Worktable'. Scan count 0, logical reads 0, physical reads

Completion time: 2021-05-17T17:08:44.2711134+03:00
```

Ελέγχοντας πάλι παραλλαγές ευρετηρίων με τις στήλες mid , userid και rating κατέληξα ότι η (mid,rating,userid) παραλλαγή έφερε τα καλύτερα αποτελέσματα.

Εντολές που χρησιμοποιήθηκαν :

```
checkpoint
dbcc dropcleanbuffers

SET statistics IO ON;

CREATE INDEX userMovieIndex ON user_movies(mid,rating,userid);

DROP INDEX userMovieIndex ON user_movies;

BEGIN
select mid, count(rating)
from user_movies group by mid order by mid

select userid, count(rating)
from user_movies group by userid order by userid
END
```

Ζήτημα Δεύτερο

Άσκηση 1

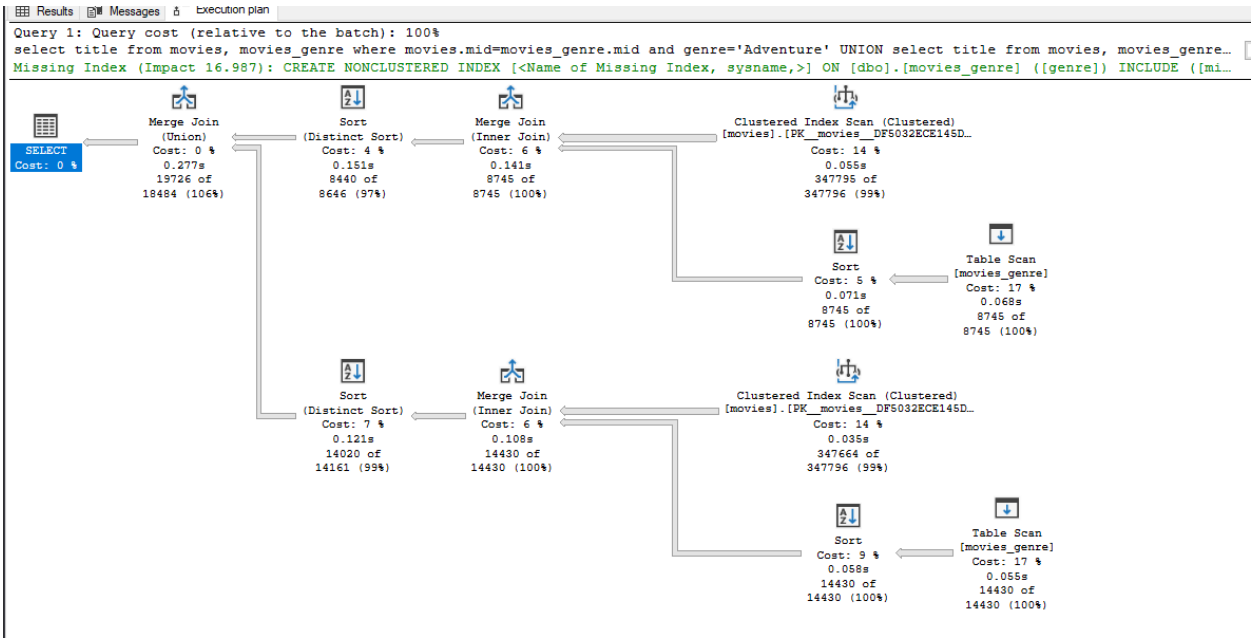
Η επερώτηση που βρήκα να βγάζει τα ίδια αποτελέσματα (με διαφορετική σειρά) και σε καλύτερο χρόνο ήταν η εξής :

```sql
SELECT title
FROM movies,movies_genre
WHERE movies.mid = movies_genre.mid AND (genre = 'Action' OR genre = 'Adventure')
GROUP BY title;
```

Στατιστικά και πλάνο εκτέλεσης επερώτησης με UNION :

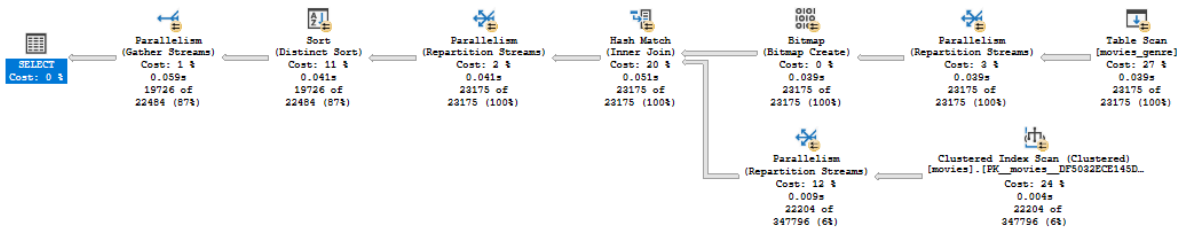
Results | Messages | Execution plan

```
(19726 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page s
Table 'movies_genre'. Scan count 2, logical reads 4416, physical reads 0,
Table 'movies'. Scan count 2, logical reads 3836, physical reads 2, page s
```



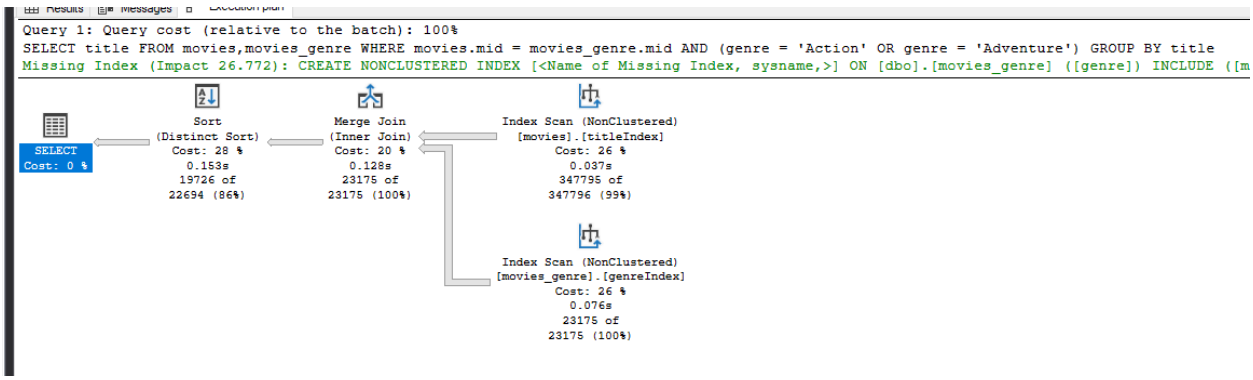
Στατιστικά και πλάνο εκτέλεσης βελτιωμένης επερώτησης :

```
(19726 rows affected)
Table 'movies_genre'. Scan count 5, logical reads 2208, physical reads 0, page
Table 'movies'. Scan count 5, logical reads 2012, physical reads 0, page server
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server
```



Δημιούργησα τα επόμενα 2 indexes και παρατήρησα επιτάχυνση του επερωτήματός μου (η λογική είναι ότι τα index ψάχνουν στην στήλη του WHERE ΚΑΙ του SELECT / GROUP BY δίνοντας μας το αποτέλεσμα γρηγορότερα από ότι με clustered index scan ή table scan , το βλέπουμε από τα logical reads) :

```
(19726 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page
Table 'movies_genre'. Scan count 1, logical reads 1375, physical reads 0,
Table 'movies'. Scan count 1, logical reads 1403, physical reads 0, page
```



Εντολές που χρησιμοποιήθηκαν :

```sql
/* Ζήτημα Δεύτερο */
/* Άσκηση 1*/


checkpoint
dbcc dropcleanbuffers

SET statistics IO ON;

CREATE INDEX titleIndex ON movies(mid,title);
CREATE INDEX genreIndex ON movies_genre(mid,genre);

DROP INDEX titleIndex ON movies;
DROP INDEX genreIndex ON movies_genre;

SELECT title
FROM movies,movies_genre
WHERE movies.mid = movies_genre.mid AND (genre = 'Action' OR genre = 'Adventure')
GROUP BY title;
```

Άσκηση 2

Τα επερωτήματα που χρησιμοποίησα είναι τα εξής :

```sql
SELECT DISTINCT title, movies.mid
FROM movies,actors,roles
WHERE movies.mid = roles.mid AND actors.aid = roles.aid AND gender = 'M'
EXCEPT
SELECT DISTINCT title,movies.mid
FROM movies,roles,actors
WHERE movies.mid = roles.mid AND actors.aid = roles.aid AND gender = 'F';
```

```
(41819 rows affected)
Table 'movies'. Scan count 10, logical reads 4024, physical reads 0, page server reads 0, re
Table 'roles'. Scan count 10, logical reads 8978, physical reads 0, page server reads 0, rea
Table 'actors'. Scan count 10, logical reads 7060, physical reads 0, page server reads 0, re
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, reac
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, rea
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page server reads 0, rea
```

```sql
SELECT DISTINCT title,movies.mid
FROM movies,actors,roles
WHERE movies.mid = roles.mid AND actors.aid = roles.aid AND gender = 'M' AND movies.mid NOT IN (
    SELECT DISTINCT movies.mid
    FROM movies,actors,roles
    WHERE movies.mid = roles.mid AND actors.aid = roles.aid AND gender = 'F'
);
```

```
(41819 rows affected)
Table 'roles'. Scan count 10, logical reads 8978, physical reads 0, pa
Table 'actors'. Scan count 10, logical reads 7060, physical reads 0, p
Table 'movies'. Scan count 10, logical reads 4024, physical reads 0, p
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, pa
```

Τα ευρετήρια που χρησιμοποιήθηκαν ήταν τα εξής :

```sql
CREATE INDEX titleIdIndex ON movies(title,mid);
CREATE INDEX genderIndex ON actors(gender);
CREATE INDEX roleIndex ON roles(aid);
```

Για το 1ο επερώτημα με indexes τα αποτελέσματα ήταν τα εξής :

```
(41819 rows affected)
Table 'movies'. Scan count 10, logical reads 2952, physical reads 0,
Table 'roles'. Scan count 10, logical reads 3826, physical reads 0,
Table 'actors'. Scan count 10, logical reads 1177, physical reads 0,
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, p
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0,
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0,
```

Για το 2ο επερώτημα με indexes τα αποτελέσματα ήταν τα εξής :

```
(41819 rows affected)
Table 'roles'. Scan count 10, logical reads 3826, pl
Table 'actors'. Scan count 10, logical reads 1177,
Table 'movies'. Scan count 10, logical reads 2952,
Table 'Worktable'. Scan count 0, logical reads 0, pl
Table 'Worktable'. Scan count 0, logical reads 0, pl
```

Παρατηρούμε ότι τα αποτελέσματα με ευρετήρια είναι σαφώς καλύτερα από ότι χωρίς , παρ'όλα αυτά τα επερωτήματα έχουν τις ίδιες αποδόσεις. Οποιοδήποτε από τα 2 κι αν επιλέξει κάποιος θα έχει τα αποτελέσματα που θέλει εξίσου γρήγορα.

Ζήτημα Τρίτο

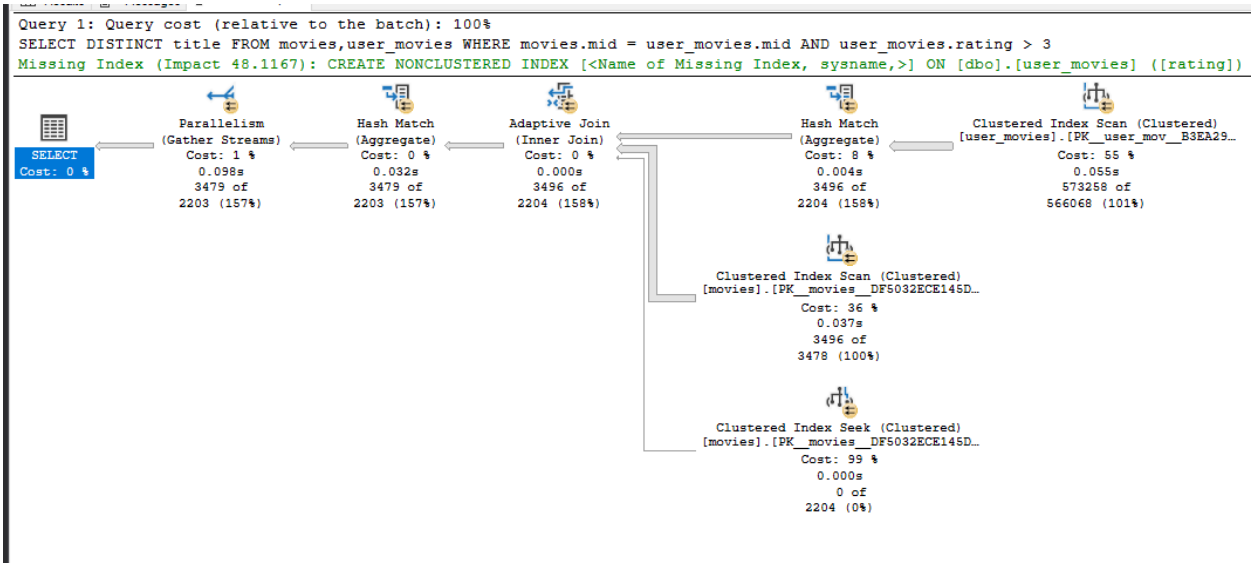
Άσκηση 1

Πρώτη επερώτηση :

«Εμφάνισε όλους τους τίτλους ταινιών με βαθμολογία μεγαλύτερη του 3.»

```
/* Ζήτημα Τρίτο */
/* Άσκηση 1 */

SELECT DISTINCT title
FROM movies,user_movies
WHERE movies.mid = user_movies.mid AND user_movies.rating > 3;
```

```
(3479 rows affected)
Table 'movies'. Scan count 5, logical reads 2012, physical reads 2
Table 'user_movies'. Scan count 5, logical reads 2733, physical re
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0
```

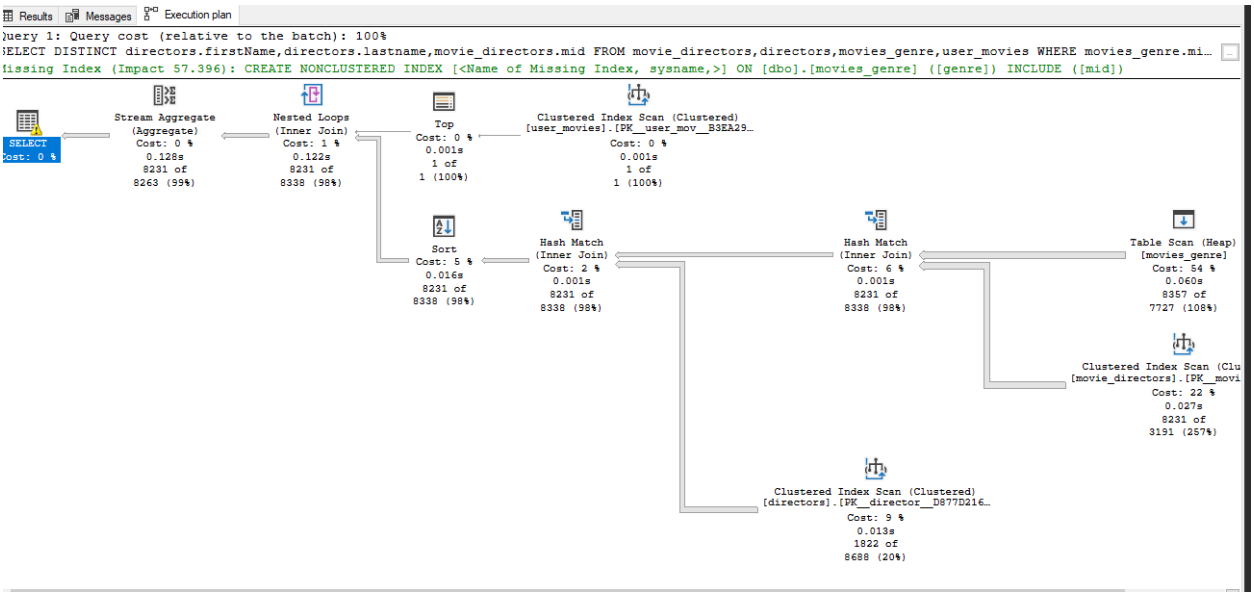


Δεύτερη επερώτηση :

«Εμφάνισε όλα τα ονοματεπώνυμα σκηνοθετών (και τα αντίστοιχα id ταινιών) που έχουν σκηνοθετήσει ταινίες με κατηγορία (genre) Western».

```
SELECT DISTINCT directors.firstName,directors.lastname,movie_directors.mid
FROM movie_directors,directors,movies_genre,user_movies
WHERE movies_genre.mid = movie_directors.mid AND directors.did = movie_directors.did AND genre = 'Western';
```

```
(8231 rows affected)
Table 'directors'. Scan count 1, logical reads 351, physical reads 1, pag
Table 'movie_directors'. Scan count 1, logical reads 675, physical reads
Table 'movies_genre'. Scan count 1, logical reads 2208, physical reads 0,
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page
Table 'user_movies'. Scan count 1, logical reads 3, physical reads 3, pag
```



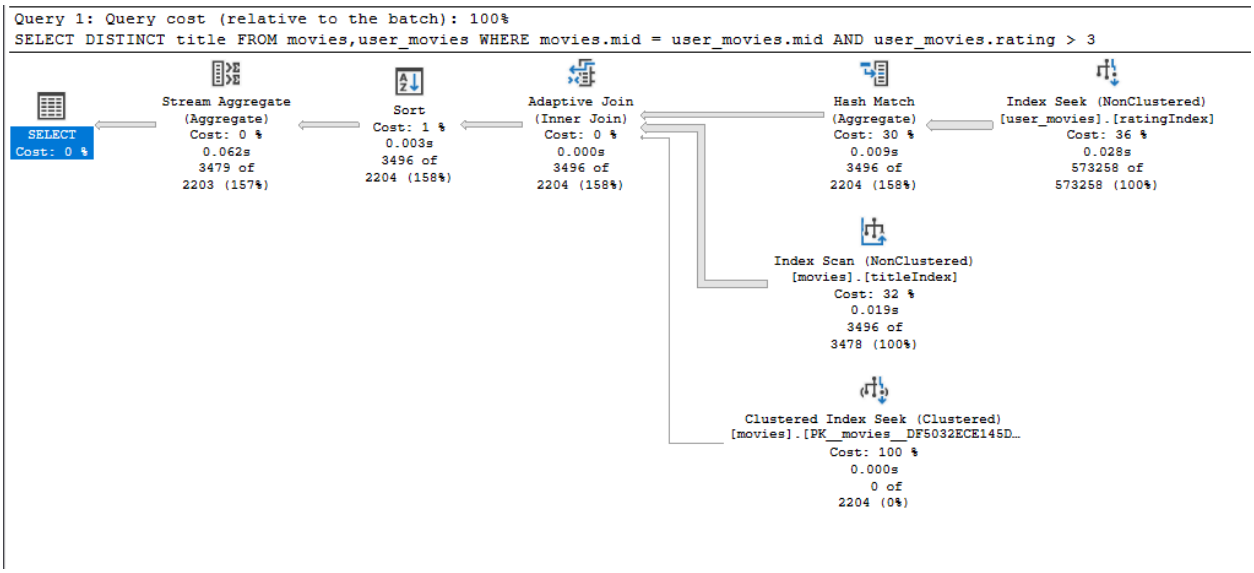
Άσκηση 2

Τα καλύτερα ευρετήρια για την πρώτη ήταν τα εξής :

```
CREATE INDEX ratingIndex ON user_movies(rating,mid);
CREATE INDEX titleIndex ON movies(title,mid);
```

Βελτιώνουν αρκετά την απόδοση των logical reads στους 2 πίνακες που θέλουμε :

```
(3479 rows affected)
Table 'movies'. Scan count 1, logical reads 1404, physical reads 0, page
Table 'user_movies'. Scan count 1, logical reads 1291, physical reads 0,
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page
```



Τα καλύτερα ευρετήρια για την δεύτερη ήταν τα εξής τέσσερα τα οποία καλύπτουν όλους τους πίνακες και επιταχύνουν αρκετά το επερώτημα (ΕΙΔΙΚΑ τα logical reads του movies_genre):

```
CREATE INDEX userTitleIndex ON user_movies(rating);
CREATE INDEX movieDirIndex ON movie_directors(did);
CREATE INDEX directorIndex ON directors(firstName,lastname);
CREATE INDEX movieGenreIndex ON movies_genre(genre,mid);
```

```
(8231 rows affected)
Table 'directors'. Scan count 1, logical reads 324, physical reads 0, page
Table 'movie_directors'. Scan count 1, logical reads 558, physical reads 0,
Table 'movies_genre'. Scan count 1, logical reads 34, physical reads 0, pag
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, page se
Table 'user_movies'. Scan count 1, logical reads 3, physical reads 0, page
```

Query 1: Query cost (relative to the batch): 100%
SELECT DISTINCT directors.firstName,directors.lastname,movie_directors.mid FROM movie_directors,directors,movies_genre,user_movies WHERE movies_genre..