

Напредни бази на податоци
Фаза 4- Индекси и оптимизација на
прашалници
Проект: VidiDB

Дамјан Димовски 231166

Кристијан Атанасов 231063

View 1: Top rated content

1. Примарен филтер за погледот `view_topRatedContent` ќе биде според просечниот рејтинг (`avg_rating`), а дополнително ќе се филтрира и според типот на содржината (`Movie` или `Series`) и насловот на медиумот.

2. Примарен случај на употреба ќе биде за прикажување на најпопуларните и најдобро оценетите содржини на почетната страна на корисничкиот интерфејс. За овој поглед ни се изклучително важни перформансите, бидејќи ова е првото нешто што корисникот го гледа при вклучување на сервисот.

3. Иницијалното време за извршување на погледот е 13s

Statistics 1				
Name	Value			
Updated Rows	0			
Execute time	13s			
Start time	Tue May 05 21:53:19 CEST 2026			
Finish time	Tue May 05 21:53:32 CEST 2026			
Query	CREATE OR REPLACE VIEW view_top_rated_content_test1 AS SELECT m.ContentID, m.title, m.releaseDate, m.AgeRating,			

Ова не е прифатливо време за апликацијата, па затоа пристапуваме кон индексирање.

4. Најбавни операции: идентификувана преку EXPLAIN ANALYZE е Full Scan (Seq Scan) на табелата Rating.

Results 1 X

EXPLAIN ANALYZE SELECT ContentContentID, ... Enter a SQL expression to filter results (use Ctrl+Space)

Grid

AZ QUERY PLAN

1 Finalize GroupAggregate (cost=309572.75..309697.87 rows=489 width=36) (actual time=235135.600..235148.311 rows=489)

2 Group Key: contentcontentid

3 -> Gather Merge (cost=309572.75..309686.86 rows=978 width=36) (actual time=235135.529..235147.748 rows=1467 loops=1)

4 Workers Planned: 2

5 Workers Launched: 2

6 -> Sort (cost=308572.73..308573.95 rows=489 width=36) (actual time=234080.315..234080.366 rows=489 loops=3)

7 Sort Key: contentcontentid

8 Sort Method: quicksort Memory: 55kB

9 Worker 0: Sort Method: quicksort Memory: 55kB

10 Worker 1: Sort Method: quicksort Memory: 55kB

11 -> Partial HashAggregate (cost=308546.00..308550.89 rows=489 width=36) (actual time=234079.936..234080.066 rows=489)

12 Group Key: contentcontentid

13 Batches: 1 Memory Usage: 121kB

14 Worker 0: Batches: 1 Memory Usage: 121kB

15 Worker 1: Batches: 1 Memory Usage: 121kB

16 -> Parallel Seq Scan on rating (cost=0.00..257608.33 rows=10187533 width=8) (actual time=304.950..231805.125 rows=10187533)

17 Planning Time: 799.098 ms

18 JIT:

19 Functions: 21

20 Options: Inlining false, Optimization false, Expressions true, Deforming true

21 Timing: Generation 2.494 ms (Deform 0.970 ms), Inlining 0.000 ms, Optimization 1.992 ms, Emission 29.263 ms, Total 33.744 ms

22 Execution Time: 235185.919 ms

Value X

Text

Finalize GroupAggregate (cost=309572.75..309697.87 rows=489 width=36) (actual time=235135.600..235148.311 rows=489)

Refresh Save Cancel

Export data 200 22

22 row(s) fetched - 3m 56s, on 2026-05-05 at 22:23:19

Error Log Query Manager X

Type query part to search in query history

Time	Type	Text	Duration	Rows	Result	Data Source	Conn
May-05 22:2...	SQL / User	EXPLAIN ANALYZE SELECT ContentContentID, AVG(RatingValue) FROM ...	3m 56s	22	Success	advdb_202526l_prj_vidid...	SQL

Базата мора да ги прочита сите записи за да го пресмета просекот, што при голем број корисници ја блокира меморијата. Времето на операциите **insert** и **update** пред индексирање изнесува:

Time	Type	Text	Duration
May-05 22:2...	SQL / User	INSERT INTO Rating (Rating_Date, RatingValue, UserUserID, ContentCont...	3.802s

5. Оптимизација и резултати По креирање на индекс на колоната ContentContentID во табелата Rating, пристапуваме кон повторно мерење.

```
CREATE INDEX idx_rating_content_id ON Rating(ContentContentID);
```

Results 1

EXPLAIN ANALYZE SELECT ContentContentID, AVG(RatingValue) FROM ...

Grid

1 Finalize GroupAggregate (cost=309572.28..309697.39 rows=489 width=36) (actual time=2966.471..2976.326 rows=489 loops=1)

2 Group Key: contentcontentid

3 -> Gather Merge (cost=309572.28..309686.39 rows=978 width=36) (actual time=2966.430..2975.797 rows=1467 loops=1)

4 Workers Planned: 2

5 Workers Launched: 2

6 -> Sort (cost=308572.26..308573.48 rows=489 width=36) (actual time=2873.288..2873.320 rows=489 loops=3)

7 Sort Key: contentcontentid

8 Sort Method: quicksort Memory: 55kB

9 Worker 0: Sort Method: quicksort Memory: 55kB

10 Worker 1: Sort Method: quicksort Memory: 55kB

11 -> Partial HashAggregate (cost=308545.53..308550.42 rows=489 width=36) (actual time=2872.991..2873.090 rows=489 loops=1)

12 Group Key: contentcontentid

13 Batches: 1 Memory Usage: 121kB

14 Worker 0: Batches: 1 Memory Usage: 121kB

15 Worker 1: Batches: 1 Memory Usage: 121kB

16 -> Parallel Seq Scan on rating (cost=0.00..257608.02 rows=10187502 width=8) (actual time=0.069..818.164 rows=815 loops=1)

17 Planning Time: 0.522 ms

18 JIT:

19 Functions: 21

20 Options: Inlining false, Optimization false, Expressions true, Deforming true

21 Timing: Generation 2.266 ms (Deform 0.923 ms), Inlining 0.000 ms, Optimization 1.525 ms, Emission 25.728 ms, Total 29.518 ms

22 Execution Time: 2977.347 ms

Value

Text

Finalize GroupAggregate (cost=309572.28..309697.39 rows=489 width=36) (actual time=2966.471..2976.326 rows=489 loops=1)

Refresh

Save

Cancel

Export data

200

22

22 row(s) fetched - 2.983s, on 2026-05-05 at 22:35:16

Error Log

Query Manager

Type query part to search in query history

Time	Type	Text	Duration	Rows	Result	Data Source
May-05 22:3...	SQL / User	EXPLAIN ANALYZE SELECT ContentContentID, AVG(RatingValue) FROM ...	2.997s	22	Success	advdb_2025261_prj_vidid...

Времето изминато во извршување на query-то со индекси изнесува **3s**, што е драстично подобрување во споредба со првичните 3m 56s. Иако базата е обемна, ова време е сега во рамките на прифатливото за генерирање на извештаи за топ содржини.

6. Време на операции по индексирање Времето изминато во извршување на операциите **insert** и **update** по индексирање изнесува:

Time	Type	Text	Duration
May-05 22:3...	SQL / User	INSERT INTO Rating (Rating_Date, RatingValue, UserUserID, ContentCont...	0.555s

View 2: User Watch History

1. Примарен филтер за погледот view_user_watch_history ќе биде според UserID на корисникот, а дополнително ќе се филтрира и според датумот на гледање (WatchedAt) и типот на уредот (DeviceType).

2. Примарен случај на употреба е кога корисникот сака да го види својот профил и листата на содржини што ги гледал претходно за да продолжи со гледање. Брзината на овој поглед директно влијае на корисничкото искуство при вчитување на почетниот екран.

3. Иницијалното време за извршување на погледот (на пример за еден корисник) изнесува:

Time	Type	Text	Duration	Rows
May-06 09:3...	SQL / User	SELECT * FROM view_user_watch_history WHERE UserID = 7288341	5.153s	200

4. Најбавните операции при ова извршување се **full scan** (Seq Scan) на табелата WatchHistory. Бидејќи табелата со историја расте многу брзо, базата троши време на пребарување на сите записи за да ги филтрира само оние за специфичен корисник.

Grid

1 Nested Loop Left Join (cost=1001.39..126674.99 rows=1 width=86) (actual time=103050.491..103050.616 rows=0 loops=1)

2 -> Nested Loop Left Join (cost=1000.97..126666.55 rows=1 width=52) (actual time=103050.490..103050.614 rows=0 loops=1)

3 -> Nested Loop (cost=1000.70..126662.23 rows=1 width=52) (actual time=103050.489..103050.611 rows=0 loops=1)

4 Join Filter: (m.contentid = wh.contentcontentid)

5 -> Index Scan using media_pkey on media m (cost=0.27..33.61 rows=489 width=22) (actual time=0.116..266.432 rows=489)

6 -> Materialize (cost=1000.42..126621.29 rows=1 width=34) (actual time=210.192..210.192 rows=0 loops=489)

7 -> Nested Loop (cost=1000.42..126621.28 rows=1 width=34) (actual time=102783.945..102784.066 rows=0 loops=1)

8 -> Gather (cost=1000.00..126612.83 rows=1 width=20) (actual time=102783.943..102784.062 rows=0 loops=1)

9 Workers Planned: 2

10 Workers Launched: 2

11 -> Parallel Seq Scan on watchhistory wh (cost=0.00..125612.73 rows=1 width=20) (actual time=102233.046..102233.046 rows=0 loops=1)

12 Filter: (userid = 1)

13 Rows Removed by Filter: 3333333

14 -> Index Scan using "User_pkey" on "User" u (cost=0.42..8.44 rows=1 width=18) (never executed)

15 Index Cond: (userid = 1)

16 -> Index Only Scan using movie_pkey on movie mo (cost=0.27..4.31 rows=1 width=4) (never executed)

17 Index Cond: (movieid = m.contentid)

18 Heap Fetches: 0

19 -> Index Scan using devices_pkey on devices d (cost=0.42..8.44 rows=1 width=14) (never executed)

20 Index Cond: (deviceid = wh.devicesdeviceid)

21 Planning Time: 2083.684 ms

22 JIT:

23 Functions: 30

24 Options: Inlining false, Optimization false, Expressions true, Deforming true

25 Timing: Generation 2.941 ms (Deform 1.407 ms), Inlining 0.000 ms, Optimization 2.173 ms, Emission 32.625 ms, Total 37.739 ms

Value

Nested Loop Left Join (cost=100

Refresh

Save

Cancel

Export data

200

26

26 row(s) fetched - 1m 47s, on 2026-05-06 at 09:32:03

Error Log

Query Manager

Type query part to search in query history

Time	Type	Text	Duration	Rows	Result	Data Source
May-06 09:3...	SQL / User	EXPLAIN ANALYZE SELECT * FROM view_user_watch_history T WHERE U...	1m 47s	26	Success	advdb_202526l_prj_vidid...

Времето на операциите **insert** и **update** пред индексирање изнесува:

Time	Type	Text	Duration	Rows
May-06 09:3...	SQL / User	INSERT INTO WatchHistory (WatchedAt, Progress_percentage, UserUserl...	4.506s	1

5. Времето изминато во извршување на query-то со индекси изнесува околу 0.02s.

За да ги подобриме перформансите, го креираме следниот индекс на надворешниот клуч:

CREATE INDEX idx_watchhistory_user_id **ON** WatchHistory(UserUserID);

EXPLAIN ANALYZE SELECT * FROM view_user_v | Enter a SQL expression to filter results (use Ctrl+Space)

Grid	Text
1	Nested Loop Left Join (cost=7.55..27.74 rows=1 width=86) (actual time=0.103..0.106 rows=0 loops=1)
2	-> Nested Loop Left Join (cost=7.13..19.30 rows=1 width=52) (actual time=0.103..0.105 rows=0 loops=1)
3	-> Nested Loop (cost=6.86..14.97 rows=1 width=52) (actual time=0.102..0.104 rows=0 loops=1)
4	-> Merge Join (cost=6.43..6.52 rows=1 width=38) (actual time=0.102..0.103 rows=0 loops=1)
5	Merge Cond: (m.contentid = wh.contentcontentid)
6	-> Index Scan using media_pkey on media m (cost=0.27..33.61 rows=489 width=22) (actual time=0.015..0.015 rows=0 loops=1)
7	-> Sort (cost=4.46..4.47 rows=1 width=20) (actual time=0.085..0.086 rows=0 loops=1)
8	Sort Key: wh.contentcontentid
9	Sort Method: quicksort Memory: 25kB
10	-> Index Scan using idx_watchhistory_user_id on watchhistory wh (cost=0.43..4.45 rows=1 width=20) (actual time=0.085..0.086 rows=0 loops=1)
11	Index Cond: (useruserid = 1)
12	-> Index Scan using "User_pkey" on "User" u (cost=0.42..8.44 rows=1 width=18) (never executed)
13	Index Cond: (userid = 1)
14	-> Index Only Scan using movie_pkey on movie mo (cost=0.27..4.31 rows=1 width=4) (never executed)
15	Index Cond: (movieid = m.contentid)
16	Heap Fetches: 0
17	-> Index Scan using devices_pkey on devices d (cost=0.42..8.44 rows=1 width=14) (never executed)
18	Index Cond: (deviceid = wh.devicesdeviceid)
19	Planning Time: 0.974 ms
20	Execution Time: 0.180 ms

Refresh Save Cancel Export data 200 20 20 row(s) fetched - 0.018s, on

Error Log Query Manager					
Type query part to search in query history					
Time	Type	Text	Duration	Rows	Result
May-06 09:4...	SQL / User	EXPLAIN ANALYZE SELECT * FROM view user watch history WHERE Us...	0.02s	20	Success

6. Времето изминато во извршување на операциите insert и update по индексирање изнесува:

Time	Type	Text	Duration	Rows	Result
May-06 09:4...	SQL / User	INSERT INTO WatchHistory (WatchedAt, Progress_percentage, UserUserI...	0.304s	1	Success

View 3: view_content_details

1. Примарен филтер за погледот view_content_details ќе биде според ContentID (ID на филмот или серијата) кога корисникот кликува на одредена содржина за да види детали. Исто така, често ќе се филтрира по наслов (title) и жанр (genres).

2. Примарен случај на употреба е прикажување на деталната страна за секој филм или серија (глумци, жанрови, времетраење). Поради големиот број на поврзувања (7+ табели), овој поглед може значително да ја забави апликацијата ако не е оптимизиран.

3. Иницијалното време за извршување на погледот за една специфична содржина изнесува:

Time	Type	Text	Duration	Rows	Result
May-06 12:4...	SQL / User	SELECT * FROM view_content_details WHERE ContentID = 501	12m 47s		Success

4. Најбавните операции се идентификувани како Sequential Scans на помошните табели за врски (Content_Genre, Content_Artist, Content_Language). Бидејќи овие табели содржат многу записи за секој медиум, базата троши време на пребарување на сите врски.

Time	Type	Text	Duration	Rows	Result	Data Source
May-06 23:3...	SQL / User	EXPLAIN ANALYZE SELECT * FROM view_content_details WHERE Conte...	10h 46m 8s		Success	advdb_2025261_prj_vidid...

5. Времето изминато по оптимизацијата со индекси:

CREATE INDEX idx_content_genre_id ON Content_Genre(ContentContentID);

CREATE INDEX idx_content_artist_id ON Content_Artist(ContentContentID);

CREATE INDEX idx_content_language_id ON Content_Language(ContentContentID);

DBeaver 26.0.1 - <advdb_202520_pgj_viddb> Script-5

File Edit Navigate Search SQL Editor Database Window Help

Database Navigator Projects

advdb_202520_pgj_viddb (194.148.125.130:5432)

advdb_202520_pgj_viddb

Schemas

Event Triggers

Extensions

Storage

System Info

Index

Administrator

System Info

Files - General

Name DataSource

Results 1

Enter a SQL expression to filter results (use Ctrl+Space)

QUERY PLAN

1 Subquery Scan on view_content_details (cost=68502.70..692516.83 rows=1 width=206) (actual time=2418367.035..2418367.055 rows=1 loops=1)

2 -> GroupAggregate (cost=68502.70..692516.82 rows=1 width=210) (actual time=2418367.028..2418367.044 rows=1 loops=1)

3 Group Key: mo.movieid, sr.totalearnings, g.name

4 Sort (cost=68502.70..685781.11 rows=299364 width=2114) (actual time=2417508.143..2417642.243 rows=300000 loops=1)

5 Sort Key: mo.movieid, sr.totalearnings, g.name

6 Sort Method: external merge, Disk: 2560288

7 Nested Loop Left Join (cost=579.25..109323.23 rows=299364 width=2114) (actual time=1638.152..2417199.149 rows=300000 loops=1)

8 -> Hash Left Join (cost=20.64..80.48 rows=6 width=2106) (actual time=1336.186..1605.104 rows=6 loops=1)

9 Hash Cond: (ca.artistid = a.artistid)

10 -> Nested Loop Left Join (cost=9.51..49.34 rows=6 width=1078) (actual time=1196.872..1445.769 rows=6 loops=1)

11 -> Nested Loop Left Join (cost=5.15..53.11 rows=2 width=502) (actual time=1077.442..1308.837 rows=2 loops=1)

12 -> Nested Loop Left Join (cost=4.95..40.82 rows=2 width=50) (actual time=1063.406..1200.740 rows=2 loops=1)

13 -> Nested Loop Left Join (cost=0.70..32.28 rows=1 width=46) (actual time=1004.814..1004.831 rows=1 loops=1)

14 -> Nested Loop Left Join (cost=0.43..23.98 rows=1 width=42) (actual time=724.057..724.067 rows=1 loops=1)

15 -> Nested Loop Left Join (cost=0.27..15.30 rows=1 width=38) (actual time=723.972..723.980 rows=1 loops=1)

16 -> Index Scan using media_play on media m (cost=0.27..8.29 rows=1 width=38) (actual time=0.877..0.882 rows=1 loops=1)

17 Index Cond: (contentid = 1)

18 -> Seq Scan on movie mo (cost=0.00..7.50 rows=1 width=8) (actual time=723.074..723.074 rows=1 loops=1)

19 Filter: (movieid = 1)

20 -> Index Scan using series_gkey on series sr (cost=0.15..8.17 rows=1 width=8) (actual time=0.071..0.071 rows=0 loops=1)

21 Index Cond: (seriesid = 1)

22 -> Index Only Scan using content_artist_play on content_artist ca (cost=0.27..8.29 rows=1 width=8) (actual time=28 rows=1 loops=1)

23 Index Cond: (contentcontentid = 1)

24 Heap Fetches: 1

25 -> Bitmap Heap Scan on content_language cl (cost=4.29..8.52 rows=2 width=8) (actual time=58.529..215.846 rows=2 loops=1)

26 Recheck Cond: (contentcontentid = 1)

27 Heap Blocks: exact=2

28 -> Bitmap Index Scan on idx_content_language_id (cost=0.00..4.29 rows=2 width=0) (actual time=12.494..12.494 rows=2 loops=1)

29 Index Cond: (contentcontentid = 1)

30 -> Memoize (cost=0.15..8.17 rows=1 width=520) (actual time=52.762..52.763 rows=1 loops=2)

31 Cache Key: cl.languageid

32 Cache Mode: logical

33 Hits: 0 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 148

34 -> Index Scan using language_gkey on language l (cost=0.14..8.16 rows=1 width=520) (actual time=52.738..52.738 rows=1 loops=2)

35 Index Cond: (languageid = cl.languageid)

36 -> Materialize (cost=4.37..16.16 rows=3 width=520) (actual time=59.711..59.732 rows=3 loops=2)

37 -> Hash Right Join (cost=4.37..16.15 rows=3 width=520) (actual time=119.417..119.450 rows=3 loops=1)

38 Hash Cond: (g.genreid = cg.genreid)

39 -> Seq Scan on genre g (cost=0.00..11.40 rows=140 width=520) (actual time=77.723..77.734 rows=29 loops=1)

40 -> Hash (cost=4.33..4.33 rows=3 width=8) (actual time=41.617..41.618 rows=3 loops=1)

41 Hash Cond: (g.genreid = cg.genreid)

42 Buckets: 1024 Batches: 1 Memory Usage: 948

43 -> Index Only Scan using content_genre_play on content_genre cg (cost=0.28..4.33 rows=3 width=8) (actual time=41.617..41.618 rows=3 loops=1)

44 Index Cond: (contentcontentid = 1)

45 Heap Fetches: 0

46 -> Hash (cost=10.50..10.50 rows=50 width=1038) (actual time=159.276..159.277 rows=50 loops=1)

47 Buckets: 1024 Batches: 1 Memory Usage: 1118

48 -> Seq Scan on artist a (cost=0.00..10.50 rows=50 width=1038) (actual time=159.234..159.242 rows=50 loops=1)

49 -> Materialize (cost=59.12..10585.44 rows=6894 width=12) (actual time=46.993..40265.631 rows=50000 loops=6)

50 -> Bitmap Heap Scan on rating r (cost=559.12..105465.57 rows=6894 width=12) (actual time=261.943..241537.750 rows=50000 loops=6)

51 Recheck Cond: (contentcontentid = 1)

52 Heap Blocks: exact=50000

53 -> Bitmap Index Scan on idx_rating_content_id (cost=0.00..548.04 rows=49994 width=0) (actual time=258.085..258.085 rows=5 loops=6)

54 Index Cond: (contentcontentid = 1)

55 Planning Time: 2611.418 ms

56 JIT:

57 Functions: 62

58 Options: Inlining time: Optimization time: Expressions true: Deforming true

59 Timing: Generation 5.363 ms, Inlining 2.279 ms, Optimization 354.907 ms, Emission 241.915 ms, Total 728.010 ms

60 Execution Time: 2418370.367 ms

Refresh Save Cancel Export data 200 59 59 rows(s) fetched - 40m 22s, on 2026-05-12 at 16:03:51

Database: advdb_202520_pgj_viddb

CEST en Writable Smart Insert 1:1 [74] Set 74 [3]

DBeaver 26.0.1 - <advdb_202520_pgj_viddb> Script-5

File Edit Navigate Search SQL Editor Database Window Help

Database Navigator Projects

advdb_202520_pgj_viddb (194.148.125.130:5432)

advdb_202520_pgj_viddb

Schemas

Event Triggers

Extensions

Storage

System Info

Index

Administrator

System Info

Files - General

Name DataSource

Results 1

Enter a SQL expression to filter results (use Ctrl+Space)

QUERY PLAN

22 -> Index Only Scan using content_artist_play on content_artist ca (cost=0.27..8.29 rows=1 width=8) (actual time=28 rows=1 loops=1)

23 Index Cond: (contentcontentid = 1)

24 Heap Fetches: 1

25 -> Bitmap Heap Scan on content_language cl (cost=4.29..8.52 rows=2 width=8) (actual time=58.529..215.846 rows=2 loops=1)

26 Recheck Cond: (contentcontentid = 1)

27 Heap Blocks: exact=2

28 -> Bitmap Index Scan on idx_content_language_id (cost=0.00..4.29 rows=2 width=0) (actual time=12.494..12.494 rows=2 loops=1)

29 Index Cond: (contentcontentid = 1)

30 -> Memoize (cost=0.15..8.17 rows=1 width=520) (actual time=52.762..52.763 rows=1 loops=2)

31 Cache Key: cl.languageid

32 Cache Mode: logical

33 Hits: 0 Misses: 2 Evictions: 0 Overflows: 0 Memory Usage: 148

34 -> Index Scan using language_gkey on language l (cost=0.14..8.16 rows=1 width=520) (actual time=52.738..52.738 rows=1 loops=2)

35 Index Cond: (languageid = cl.languageid)

36 -> Materialize (cost=4.37..16.16 rows=3 width=520) (actual time=59.711..59.732 rows=3 loops=2)

37 -> Hash Right Join (cost=4.37..16.15 rows=3 width=520) (actual time=119.417..119.450 rows=3 loops=1)

38 Hash Cond: (g.genreid = cg.genreid)

39 -> Seq Scan on genre g (cost=0.00..11.40 rows=140 width=520) (actual time=77.723..77.734 rows=29 loops=1)

40 -> Hash (cost=4.33..4.33 rows=3 width=8) (actual time=41.617..41.618 rows=3 loops=1)

41 Hash Cond: (g.genreid = cg.genreid)

42 Buckets: 1024 Batches: 1 Memory Usage: 948

43 -> Index Only Scan using content_genre_play on content_genre cg (cost=0.28..4.33 rows=3 width=8) (actual time=41.617..41.618 rows=3 loops=1)

44 Index Cond: (contentcontentid = 1)

45 Heap Fetches: 0

46 -> Hash (cost=10.50..10.50 rows=50 width=1038) (actual time=159.276..159.277 rows=50 loops=1)

47 Buckets: 1024 Batches: 1 Memory Usage: 1118

48 -> Seq Scan on artist a (cost=0.00..10.50 rows=50 width=1038) (actual time=159.234..159.242 rows=50 loops=1)

49 -> Materialize (cost=59.12..10585.44 rows=6894 width=12) (actual time=46.993..40265.631 rows=50000 loops=6)

50 -> Bitmap Heap Scan on rating r (cost=559.12..105465.57 rows=6894 width=12) (actual time=261.943..241537.750 rows=50000 loops=6)

51 Recheck Cond: (contentcontentid = 1)

52 Heap Blocks: exact=50000

53 -> Bitmap Index Scan on idx_rating_content_id (cost=0.00..548.04 rows=49994 width=0) (actual time=258.085..258.085 rows=5 loops=6)

54 Index Cond: (contentcontentid = 1)

55 Planning Time: 2611.418 ms

56 JIT:

57 Functions: 62

58 Options: Inlining time: Optimization time: Expressions true: Deforming true

59 Timing: Generation 5.363 ms, Inlining 2.279 ms, Optimization 354.907 ms, Emission 241.915 ms, Total 728.010 ms

60 Execution Time: 2418370.367 ms

Refresh Save Cancel Export data 200 59 59 rows(s) fetched - 40m 22s, on 2026-05-12 at 16:03:51

Database: advdb_202520_pgj_viddb

CEST en

View 4: view_active_subscriptions

1. Примарен филтер за погледот view_active_subscriptions ќе биде според статусот на претплатата (us.Status = 'Active'), а дополнително ќе се филтрира според Email на корисникот или UserID.

2. Примарен случај на употреба е проверка на правата за пристап на корисникот при најава (Login) и контрола на бројот на активни уреди. Овој поглед се извршува при секоја сесија, па затоа перформансите мора да бидат на високо ниво.

3. Иницијалното време за извршување на погледот изнесува:

Time	Type	Text	Duration	Rows	Result
May-11 14:3...	SQL / User	SELECT * FROM view_active_subscriptions WHERE Email = 'aleksandar.p...	0.672s	1	Success

Иако времето е под една секунда, при голем број истовремени најави, ова ќе го оптовари серверот.

4. Најбавните операции се идентификувани како **Sequential Scan** на табелата User_Subscription поради филтрирањето по колоната Status, како и пребројувањето на уредите преку LEFT JOIN со табелата Devices.

		AZ QUERY PLAN
1		GroupAggregate (cost=89502.25..221705.09 rows=604333 width=1130) (actual time=1251.000..2593.093 rows=600918 loops=1)
2		Group Key: u.userid, s.name, s.price, s.maxdevices, s.videoquality, us.start_date, us.end_date, us.auto_renew
3		-> Incremental Sort (cost=89502.25..202064.26 rows=604333 width=1126) (actual time=1250.970..2063.868 rows=600918 loops=1)
4		Sort Key: u.userid, s.name, s.price, s.maxdevices, s.videoquality, us.start_date, us.end_date, us.auto_renew
5		Presorted Key: u.userid
6		Full-sort Groups: 18779 Sort Method: quicksort Average Memory: 29kB Peak Memory: 29kB
7		-> Nested Loop (cost=89502.10..174869.28 rows=604333 width=1126) (actual time=1250.874..1796.683 rows=600918 loops=1)
8		-> Gather Merge (cost=89501.95..159886.49 rows=604333 width=74) (actual time=1250.761..1526.388 rows=600918 loops=1)
9		Workers Planned: 2
10		Workers Launched: 2
11		-> Sort (cost=88501.93..89131.44 rows=251805 width=74) (actual time=926.450..979.270 rows=200306 loops=3)
12		Sort Key: u.userid
13		Sort Method: external merge Disk: 17768kB
14		Worker 0: Sort Method: external merge Disk: 17776kB
15		Worker 1: Sort Method: external merge Disk: 17744kB
16		-> Parallel Hash Left Join (cost=33420.98..54723.07 rows=251805 width=74) (actual time=512.104..748.851 rows=200306 loops=1)
17		Hash Cond: (us.usersubscriptionid = d.usersubscriptionid)
18		-> Parallel Hash Join (cost=27419.07..47333.39 rows=251805 width=74) (actual time=450.821..595.068 rows=200306 loops=1)
19		Hash Cond: (us.userid = u.userid)
20		-> Parallel Seq Scan on user_subscription us (cost=0.00..12973.33 rows=251805 width=31) (actual time=0.12..0.12 rows=251805 loops=1)
21		Filter: ((status)::text = 'Active'::text)
22		Rows Removed by Filter: 133027
23		-> Parallel Hash (cost=20550.81..20550.81 rows=322581 width=47) (actual time=268.097..268.098 rows=322581 loops=1)
24		Buckets: 131072 Batches: 16 Memory Usage: 6272kB
25		-> Parallel Seq Scan on "User" u (cost=0.00..20550.81 rows=322581 width=47) (actual time=10.144..145.145 rows=322581 loops=1)
26		-> Parallel Hash (cost=3800.29..3800.29 rows=176129 width=8) (actual time=59.293..59.294 rows=99807 loops=1)
27		Buckets: 524288 Batches: 1 Memory Usage: 15840kB

Time	Type	Text	Duration	Rows	Result
May-11 14:4...	SQL / User	EXPLAIN ANALYZE SELECT * FROM view_active_subscriptions WHERE S...	2.676s	40	Success

5. Времето изминато по оптимизацијата со индекси изнесува околу 2.2ms.

Решение: Креираме индекс на колоната Status (бидејќи е дел од WHERE условот) и на надворешните клучеви за побрз JOIN.

```
CREATE INDEX idx_subscription_status ON User_Subscription(Status);
CREATE INDEX idx_devices_subscription_id ON Devices(UserSubscriptionID);
```

EXPLAIN ANALYZE SELECT * FROM view_active | Enter a SQL expression to filter results (use Ctrl+Space)

Grid	○ A-Z QUERY PLAN
1	GroupAggregate (cost=89502.25..221705.09 rows=604333 width=1130) (actual time=902.550..2216.600 rows=600918 loops=1)
2	Group Key: u.userid, s.name, s.price, s.maxdevices, s.videoquality, us.start_date, us.end_date, us.auto_renew
3	-> Incremental Sort (cost=89502.25..202064.26 rows=604333 width=1126) (actual time=902.520..1689.217 rows=600918 loops=1)
4	Sort Key: u.userid, s.name, s.price, s.maxdevices, s.videoquality, us.start_date, us.end_date, us.auto_renew
5	Presorted Key: u.userid
6	Full-sort Groups: 18779 Sort Method: quicksort Average Memory: 29kB Peak Memory: 29kB
7	-> Nested Loop (cost=89502.10..174869.28 rows=604333 width=1126) (actual time=902.439..1422.478 rows=600918 loops=1)
8	-> Gather Merge (cost=89501.95..159886.49 rows=604333 width=74) (actual time=902.289..1150.916 rows=600918 loops=1)
9	Workers Planned: 2
10	Workers Launched: 2
11	-> Sort (cost=88501.93..89131.44 rows=251805 width=74) (actual time=870.690..921.547 rows=200306 loops=3)
12	Sort Key: u.userid
13	Sort Method: external merge Disk: 17528kB
14	Worker 0: Sort Method: external merge Disk: 18256kB
15	Worker 1: Sort Method: external merge Disk: 17504kB
16	-> Parallel Hash Left Join (cost=33420.98..54723.07 rows=251805 width=74) (actual time=504.391..734.007 rows=200306 loops=1)
17	Hash Cond: (us.usersubscriptionid = d.usersubscriptionid)
18	-> Parallel Hash Join (cost=27419.07..47333.39 rows=251805 width=74) (actual time=428.472..565.293 rows=200306 loops=1)
19	Hash Cond: (us.useruserid = u.userid)
20	-> Parallel Seq Scan on user_subscription us (cost=0.00..12973.33 rows=251805 width=31) (actual time=0.083..0.083 rows=251805 loops=1)
21	Filter: ((status)::text = 'Active')::text
22	Rows Removed by Filter: 133027
23	-> Parallel Hash (cost=20550.81..20550.81 rows=322581 width=47) (actual time=258.576..258.578 rows=33333 loops=1)
24	Buckets: 131072 Batches: 16 Memory Usage: 6272kB
25	-> Parallel Seq Scan on "User" u (cost=0.00..20550.81 rows=322581 width=47) (actual time=0.069..0.137 rows=33333 loops=1)
26	-> Parallel Hash (cost=3800.29..3800.29 rows=176129 width=8) (actual time=74.018..74.019 rows=99807 loops=1)
27	Buckets: 524288 Batches: 1 Memory Usage: 15840kB
28	-> Parallel Seq Scan on devices d (cost=0.00..3800.29 rows=176129 width=8) (actual time=18.736..38.846 rows=99807 loops=1)

Record

AZ QUERY PLAN
Worker 1: Sort Method: external merge Disk: 17504kB
-> Parallel Hash Left Join (cost=33420.98..54723.07 rows=251805 width=74) (actual time=504.391..734.007 rows=200)
Hash Cond: (us.usersubscriptionid = d.usersubscriptionid)
-> Parallel Hash Join (cost=27419.07..47333.39 rows=251805 width=74) (actual time=428.472..565.293 rows=200)
Hash Cond: (us.useruserid = u.userid)
-> Parallel Seq Scan on user_subscription us (cost=0.00..12973.33 rows=251805 width=31) (actual time=0.083..0.083 rows=251805 width=31)
Filter: ((status)::text = 'Active'::text)
Rows Removed by Filter: 133027
-> Parallel Hash (cost=20550.81..20550.81 rows=322581 width=47) (actual time=258.576..258.578 rows=3333)
Buckets: 131072 Batches: 16 Memory Usage: 6272kB
-> Parallel Seq Scan on "User" u (cost=0.00..20550.81 rows=322581 width=47) (actual time=0.069..137.069 rows=3333)
-> Parallel Hash (cost=3800.29..3800.29 rows=176129 width=8) (actual time=74.018..74.019 rows=99807 loops=1)
Buckets: 524288 Batches: 1 Memory Usage: 15840kB
-> Parallel Seq Scan on devices d (cost=0.00..3800.29 rows=176129 width=8) (actual time=18.736..38.846 rows=99807)
-> Memoize (cost=0.15..0.17 rows=1 width=1060) (actual time=0.000..0.000 rows=1 loops=600918)
Cache Key: us.subscriptionsubscriptionid
Cache Mode: logical
Hits: 600917 Misses: 1 Evictions: 0 Overflows: 0 Memory Usage: 1kB
-> Index Scan using subscription_pkey on subscription s (cost=0.14..0.16 rows=1 width=1060) (actual time=0.086..0.087 rows=1)
Index Cond: (subscriptionid = us.subscriptionsubscriptionid)
Planning Time: 1.536 ms
JIT:
Functions: 75
Options: Inlining false, Optimization false, Expressions true, Deforming true
Timing: Generation 5.546 ms (Deform 2.686 ms), Inlining 0.000 ms, Optimization 2.635 ms, Emission 53.958 ms, Total 62.140 ms
Execution Time: 2246.275 ms

View 5: view_series_episodes

1. Примарен филтер за погледот view_series_episodes ќе биде според series_id (ID на серијата), со цел да се излистаат сите достапни епизоди за таа серија организирани по сезони.

2. Примарен случај на употреба е кога корисникот избира одредена серија и системот треба да ја прикаже структурата на сезоните и листата на епизоди. Ова е критична операција за корисничкиот интерфејс на секоја стриминг платформа.

3. Иницијалното време за извршување на погледот за една серија изнесува:

Time	Type	Text	Duration	Rows	Result
May-11 14:4...	SQL / User	SELECT * FROM view_series_episodes WHERE series_id = 101	0.783s	13	Success

4. Најбавните операции се идентификувани како **Sequential Scans** на табелите Season и Episode. Бидејќи базата мора да ги провери сите сезони и сите епизоди за да ги најде оние што припаѓаат на соодветната серија, времето на извршување расте линеарно со големината на податоците.

Grid

Text

Record

AZ QUERY PLAN

1

Sort (cost=78.33..78.36 rows=13 width=60) (actual time=0.846..0.850 rows=13 loops=1)

2

Sort Key: m.title, sea.seasonnumber, e.episodenum

3

Sort Method: quicksort Memory: 26kB

4

-> Nested Loop (cost=5.15..78.09 rows=13 width=60) (actual time=0.139..0.815 rows=13 loops=1)

5

-> Nested Loop (cost=0.43..16.47 rows=1 width=30) (actual time=0.046..0.050 rows=1 loops=1)

6

-> Index Scan using series_pkey on series se (cost=0.15..8.17 rows=1 width=8) (actual time=0.000..0.001 rows=1 loops=1)

7

Index Cond: (seriesid = 10)

8

-> Index Scan using media_pkey on media m (cost=0.27..8.29 rows=1 width=22) (actual time=0.000..0.001 rows=1 loops=1)

9

Index Cond: (contentid = 10)

10

-> Hash Join (cost=4.73..61.48 rows=13 width=38) (actual time=0.091..0.760 rows=13 loops=1)

11

Hash Cond: (e.seasonseasonid = sea.seasonid)

12

-> Seq Scan on episode e (cost=0.00..49.21 rows=2821 width=26) (actual time=0.000..0.001 rows=2821 loops=1)

13

-> Hash (cost=4.71..4.71 rows=1 width=16) (actual time=0.030..0.031 rows=1 loops=1)

14

Buckets: 1024 Batches: 1 Memory Usage: 9kB

15

-> Seq Scan on season sea (cost=0.00..4.71 rows=1 width=16) (actual time=0.000..0.001 rows=1 loops=1)

16

Filter: (seriesid = 10)

17

Rows Removed by Filter: 216

18

Planning Time: 0.628 ms

19

Execution Time: 0.937 ms

Refresh

Save

Cancel

</

Времето опфаќа 0.02s што е сосема прифатливо време.

View 6: view_user_watchlist — Листа на зачувани содржини (Watchlist)

1. **Примарен филтер** за погледот view_user_watchlist ќе биде според UserID на корисникот, бидејќи секогаш се прикажуваат податоци за специфичен најавен корисник. Дополнително, погледот е сортиран според dateAdded за да се прикажат најновите додадени ставки најгоре.
2. **Примарен случај на употреба** е прикажување на персонализираната листа на содржини кои корисникот планира да ги гледа. Бидејќи ова е една од основните функции на корисничкиот интерфејс, брзината на одговор е клучна за течно корисничко искуство.
3. **Иницијалното време** за извршување на погледот за еден корисник изнесува:

Time	Type	Text	Duration	Rows	Result
May-11 14:5...	SQL / User	SELECT * FROM view_user_watchlist WHERE UserID = 325637	1.553s	200	Success

4. **Најбавните операции** се идентификувани како **Sequential Scan** на табелата Watchlist. Бидејќи базата мора да ги пребара сите записи од сите корисници за да ги извлече само оние за бараниот UserID, се троши непотребно време и меморија.

Results 1 X

EXPLAIN ANALYZE SELECT * FROM view_user_1 Enter a SQL expression to filter results (use Ctrl+Space)

Grid

Text

Record

A-Z QUERY PLAN

1

Sort (cost=1753.02..1754.24 rows=487 width=80) (actual time=0.989..1.017 rows=489 loops=1)

2

Sort Key: wl.dateadded DESC

3

Sort Method: quicksort Memory: 59kB

4

-> Hash Left Join (cost=42.53..1731.28 rows=487 width=80) (actual time=0.428..0.791 rows=489 loops=1)

5

Hash Cond: (m.contentid = mo.movieid)

6

-> Hash Join (cost=30.63..1718.09 rows=487 width=52) (actual time=0.298..0.548 rows=489 loops=1)

7

Hash Cond: (wl.contentcontentid = m.contentid)

8

-> Nested Loop (cost=12.63..1698.80 rows=487 width=26) (actual time=0.074..0.202 rows=489 loops=1)

9

-> Index Scan using "User_pkey" on "User" u (cost=0.42..8.44 rows=1 width=18) (actual time=0.000..0.001 rows=1 loops=1)

10

Index Cond: (userid = 325637)

11

-> Bitmap Heap Scan on watchlist wl (cost=12.20..1685.49 rows=487 width=12) (actual time=0.000..0.001 rows=1 loops=1)

12

Recheck Cond: (useruserid = 325637)

13

Heap Blocks: exact=4

14

-> Bitmap Index Scan on watchlist_useruserid_contentcontentid_key (cost=0.00..0.00 rows=1 width=0) (actual time=0.000..0.001 rows=1 loops=1)

15

Index Cond: (useruserid = 325637)

16

-> Hash (cost=11.89..11.89 rows=489 width=30) (actual time=0.216..0.216 rows=489 loops=1)

17

Buckets: 1024 Batches: 1 Memory Usage: 39kB

18

-> Seq Scan on media m (cost=0.00..11.89 rows=489 width=30) (actual time=0.017..0.017 rows=489 loops=1)

19

-> Hash (cost=6.40..6.40 rows=440 width=4) (actual time=0.121..0.122 rows=440 loops=1)

20

Buckets: 1024 Batches: 1 Memory Usage: 24kB

21

-> Seq Scan on movie mo (cost=0.00..6.40 rows=440 width=4) (actual time=0.011..0.061 rows=440 loops=1)

22

Planning Time: 0.631 ms

23

Execution Time: 1.089 ms

Refresh

Save

Cancel

Export data

200

23

23 row(s) fetched - 0.018s, on

Error Log

Query Manager X

Type query part to search in query history

Time	Type	Text	Duration	Rows	Result
May-11 14:5...	SQL / User	EXPLAIN ANALYZE SELECT * FROM view_user_watchlist WHERE User...	0.018s	23	Success

Времето опфаќа 0.18s што е сосема прифатливо време.

Глобална Оптимизација на Клучни Табели (Big Data Optimization)

1. Идентификување на критични точки

При анализа на базата, утврдено е дека табелите Rating, WatchHistory и Review се најобемни (со над 20 милиони записи). Поради нивната големина, секоја агрегација или филтрирање преку надворешни клучеви предизвикува енормно забавување на целиот систем.

2. Имплементација на индекси на надворешни клучеви (FK Indexes)

За да се овозможи брзо поврзување на овие табели со корисниците и содржините, се имплементираа следните индекси:

-- Оптимизација на Rating (24M записи)

```
CREATE INDEX idx_rating_content ON Rating (ContentContentID);
```

```
CREATE INDEX idx_rating_user ON Rating (UserUserID);
```

-- Оптимизација на WatchHistory

```
CREATE INDEX idx_watchhistory_content ON WatchHistory (ContentContentID);
```

```
CREATE INDEX idx_watchhistory_user ON WatchHistory (UserUserID);
```

-- Оптимизација на Review

```
CREATE INDEX idx_review_content ON Review (ContentContentID);
```

```
CREATE INDEX idx_review_user ON Review (UserUserID);
```

3. Заклучок

Оваа оптимизација е клучна за скалабилноста на системот. Иако индексите зафаќаат дополнителен простор на дискот, драстичното намалување на времето на одговор (од часови во минути/секунди) овозможува непречено функционирање на аналитичките функции и извештаите во реално време.

Параметар	Пред Индексирање	По Индексирање	Подобрување
Време на извршување	~ 60 min (1 час)	1.7 min (102s)	35.2x побрзо
Скенирање податоци	Sequential Scan	Index Scan	/
Обем на податоци	24,000,000 записи	24,000,000 записи	/